

## Index

Index.....	1
1    Introduction.....	5
1.1    What is SimFlex CS?.....	5
1.2    Benefits.....	5
1.3    Applications.....	5
1.4    Key Features .....	6
1.5    Test Sessions.....	6
1.6    Test Suite .....	6
1.7    Script Edition .....	6
1.8    Logging .....	6
1.9    Basic skill requirements.....	7
1.9.1    IEC 61850 standard, tissues and UCA testing.....	7
1.9.2    Python/Programming .....	7
2    SimFlex CS overview .....	8
2.1    Details in the main screen .....	8
2.2    Configuring the SimFlex CS settings .....	8
2.3    Detailed information of Settings dialog, General tab .....	9
2.4    Detailed information of Settings dialog, Devices tab.....	10
2.5    Detailed information of Settings dialog, NIC tab.....	11
2.6    Detailed information of Settings dialog, Globals tab .....	12
2.6.1    Program settings .....	13
2.6.2    Global test suite settings .....	13
2.6.3    Settings for test cases in Conformance Block 1 .....	14
2.6.4    Settings for test cases in Conformance Block 2 .....	14
2.6.5    Settings for test cases in Conformance Block 2+.....	14
2.6.6    Settings for test cases in Conformance Block 3 .....	15

2.6.7	Settings for test cases in Conformance Block 4 .....	15
2.6.8	Settings for test cases in Conformance Block 5 & 6 .....	15
2.6.9	Settings for test cases in Conformance Block 5 .....	16
2.6.10	Settings for test cases in Conformance Block 6 .....	16
2.6.11	Settings for test cases in Conformance Block 9 .....	16
2.6.12	Common Settings for test cases in Conformance Blocks 12 .....	17
2.6.13	Settings for test cases in Conformance Block 12a.....	17
2.6.14	Settings for test cases in Conformance Block 12b .....	18
2.6.15	Settings for test cases in Conformance Block 12c.....	18
2.6.16	Settings for test cases in Conformance Block 12d .....	18
2.6.17	Settings for test cases in Conformance Block 13 .....	19
2.6.18	Settings for test cases in Conformance Block 14 .....	19
3	Use case: Starting and stopping SimFlex CS .....	19
3.1	Verify a connection with a DUT.....	19
3.2	Start SimFlex CS .....	20
3.3	Configure the right network adapter in SimFlex CS .....	20
3.4	Select, Edit or Add a DUT to SimFlex CS .....	20
3.5	Build the data model .....	20
4	Use case: Run a single test .....	21
4.1	Open a test suite .....	21
4.2	Test suite tree panel and script tabs .....	22
4.3	Select and run test scripts .....	24
4.4	Evaluate test results .....	27
4.5	Open and examine a log file .....	28
4.6	Open and examine a capture file .....	29
4.7	Save and close a test suite.....	30
5	Use case: Run a session .....	32

5.1	Create a test session.....	32
5.2	Open a test session.....	34
5.3	Run tests in ‘non-recording’ mode .....	36
5.4	Run tests in ‘recording’ mode .....	36
5.5	Save and close test sessions.....	36
5.6	Modify and copy test sessions .....	38
5.7	Use the issues tab.....	40
6	Advanced use .....	42
6.1	Global settings.....	42
6.1.1	Columns in the Global Settings Tab.....	42
6.1.2	Global variable format.....	43
6.2	More on Scripts .....	43
6.2.1	IEC 61850 and UCA test procedures.....	45
6.2.2	Script structure and language .....	45
6.2.3	Function calls .....	46
6.3	How to use the data model.....	46
6.3.1	Tree hierarchy .....	47
6.3.2	How to read the values in the DUT data model (MMSData values) .....	47
6.3.3	MMSData type and length information: .....	48
6.4	More on configuration files.....	49
6.4.1	Log file customization.....	49
6.4.2	Test Lab information .....	51
6.4.3	Vendor information.....	52
7	Frequently Asked Questions (FAQ) .....	53
7.1	General .....	53
7.1.1	How to install the SimFlex™ Client Simulator?.....	53
7.1.2	How can I check if my device is available on the network? .....	53

## SimFlex™ Client Simulator Edition 2 – User Manual

7.1.3	The Client Simulator does not start or complains about the license .....	53
7.1.4	Missing Module ‘OS’.....	54
7.2	Test suites.....	55
7.2.1	How to copy a test suite from a session?.....	55
8	Additional Information .....	55
8.1	File name conventions.....	55
8.2	Used abbreviations.....	56
8.3	Referenced documents .....	56
8.4	Referenced tools .....	56
8.5	Document Version History .....	56

## 1 Introduction

### 1.1 What is SimFlex CS?

The SimFlex IEC 61850 Client Simulator Edition 2 is a simulator for the testing of IEC 61850 based servers (IEDs) that implement the Edition 2 version of the IEC 61850 standard. This software tool is designed for performing IEC 61850 tests including Conformance Tests. The main features of the Client Simulator include:

- Script based for flexibility and expansion of the test system
- Graphical User Interface
- Automatic execution of scripts
- Report & Storage of test results

The SimFlex IEC 61850 Client Simulator is an easy to use tool with advanced capabilities that enables utilities, manufacturers, system integrators and conformance test laboratories to automatically test IEC 61850 based server devices.

The SimFlex IEC 61850 Client Simulator comes with an extensive test suite that implements the IEC 61850 test cases that can be individually selected and executed. The SimFlex™ IEC 61850 Client Simulator enables users to create new and combine existing test cases to perform complex test scenarios. The simulator is easy to use and its intuitive user interface enables to quickly perform tests.

This makes the SimFlex IEC 61850 Client Simulator an excellent software solution for the electrical power industry.

### 1.2 Benefits

The main benefits of using the SimFlex Client Simulator are:

- Fast and flexible automated testing of system and IED behavior
- Verification of IEC 61850 implementations
- Easy setup of a conformance test laboratory
- Generation and capturing of standard network traffic that can be stored in files

### 1.3 Applications

The SimFlex Client Simulator has a wide range of applications.

- IEC 61850 Edition 2 conformance testing
- Functional testing of IEDs and systems
- IED product development
- GOOSE publishing
- Network traffic generation
- Preparation for UCA© International Users Group based IED certification

## 1.4 Key Features

The SimFlex Client Simulator is designed to be a versatile simulator for IEC 61850 devices and systems. Key features of the SimFlex Client Simulator include:

- Performs the test cases from the UCA© International Users Group detailed test procedures, based on IEC61850-10 Edition 2
- Through a flexible interface any test case can be designed and executed
- Logging of test progress and test results in human-readable text format
- Network traffic is captured in PCAP format that can be read by other tools such as Wireshark
- Advanced GOOSE engine enables for publishing GOOSE messages, including faulty ones for negative testing
- Enhanced test case editor with syntax highlighting
- Visual representation of and interaction with server's data models

The SimFlex™ IEC 61850 Client Simulator provides a flexible user interface for IEC 61850 testing.

Many of the features of the program are available through an intuitive tool bar.

## 1.5 Test Sessions

Testing of IEDs and systems requires that test results can be stored orderly. Especially for IEC 61850 Conformance Test Laboratories it is important that test information and results are kept together and stored safely. The SimFlex Client Simulator provides Test Sessions that enable the user to create a specific environment for the device or system under test where all necessary information and test results are stored. This makes it possible to organize the results and easily analyze test results at a later stage.

## 1.6 Test Suite

The core of the simulator is an engine that executes predefined and user defined test definitions. These definitions are located in groups that, together, form a test suite. The test suite is represented as a hierarchical tree.

The test suite view offers functionality to add, rename and delete groups and test scripts. Each group and test script can be enabled separately, which makes it possible to execute any combination of groups and test scripts.

To facilitate its use the SimFlex Client Simulator comes with a complete test suite that implements the UCA© International Users Group detailed test procedures for server devices.

## 1.7 Script Edition

The SimFlex Client Simulator has a built-in editor for creating the test definitions to be executed. The interface provides many of the defined services in IEC 61850-7-2 to be called from within the test sequence. The response from the called services can be checked and the flow of the test scripts can be logged in the Logging View. With a simple command the user can start and stop network capturing and store the captured communication in a PCAP file format. It is also possible to define and show message boxes when user-input is required in the script.

## 1.8 Logging

In this view the user can follow the execution of the scripts. Log messages from the test sequence as well as messages from the test engine are shown.

The SimFlex Client Simulator automatically shows the response and retuned values of called IEC 61850 services. This makes it very easy for the user to monitor test execution.

## **1.9 Basic skill requirements**

There are some main things that the user must be familiarized beforehand, in order to use this program.

### **1.9.1 IEC 61850 standard, tissues and UCA testing.**

The user/tester of this application must have at least basic level understanding of IEC 61850 standards.

This is important because the tester will use it to test the device, diagnose and analyze the result. The user must be correctly identify that each of the network communication between the server/device and the client is exactly the way it should, given the scope within the IEC 61850 standards.

### **1.9.2 Python/Programming**

The user/tester at least must understand how to do basic programming. It will be better if the user/tester is already familiar with Python programming. The SimFlex Client Simulator uses Python as the programming language for the test cases because it is a high level programming language, easily understandable and configurable.

## 2 SimFlex CS overview

### 2.1 Details in the main screen

The main window has different parts: the menu bar (1), the toolbar (2), the test suite panel (3), the script panel (4), the device panel (5) and the log/issues panel (6).

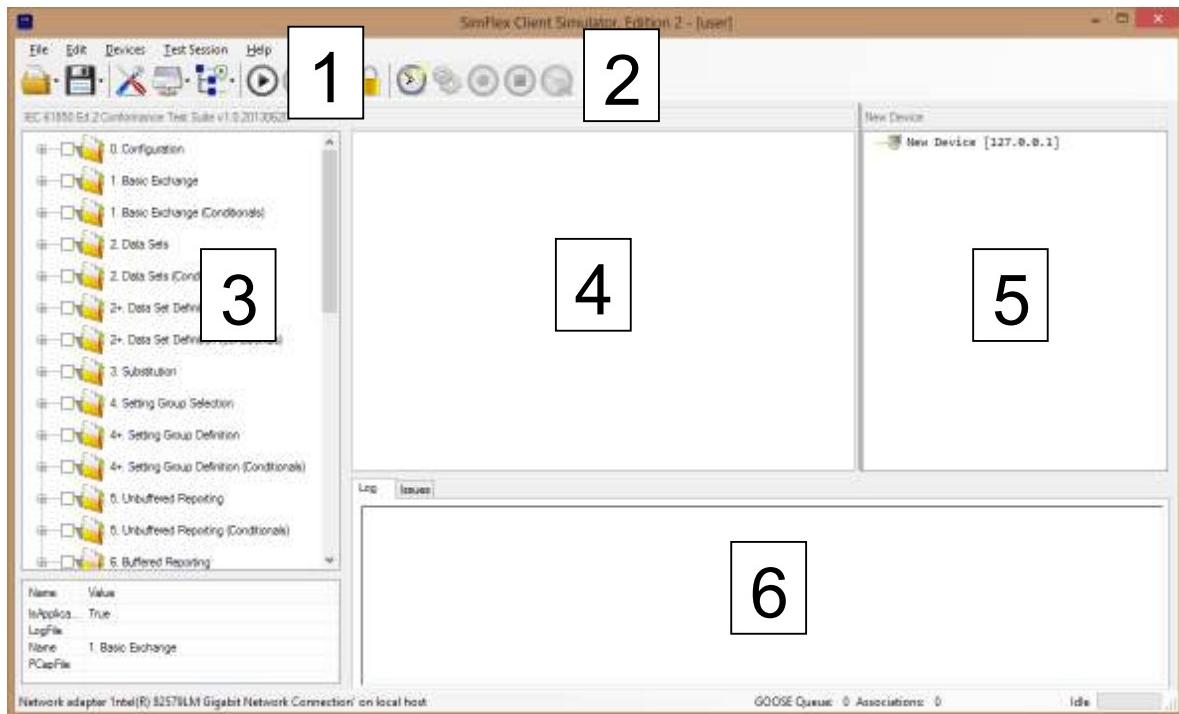


Figure 2.01

### 2.2 Configuring the SimFlex CS settings

In order to configure the setting:

- 1) In the menu bar click in **Edit → Settings**

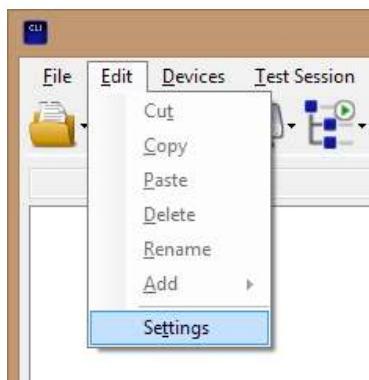


Figure 2.02

- 2) In the toolbar, hover above the icon that will says "show settings" and click the icon.

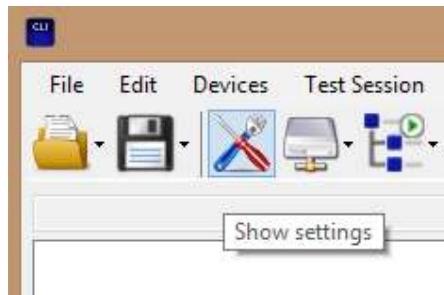


Figure 2.03

There will a new window shows up. The window consists of four tabs.

### 2.3 Detailed information of Settings dialog, General tab

In the Settings dialog, General tab, you can specify program specific settings. The following figure shows the settings that can be configured in the General tab.

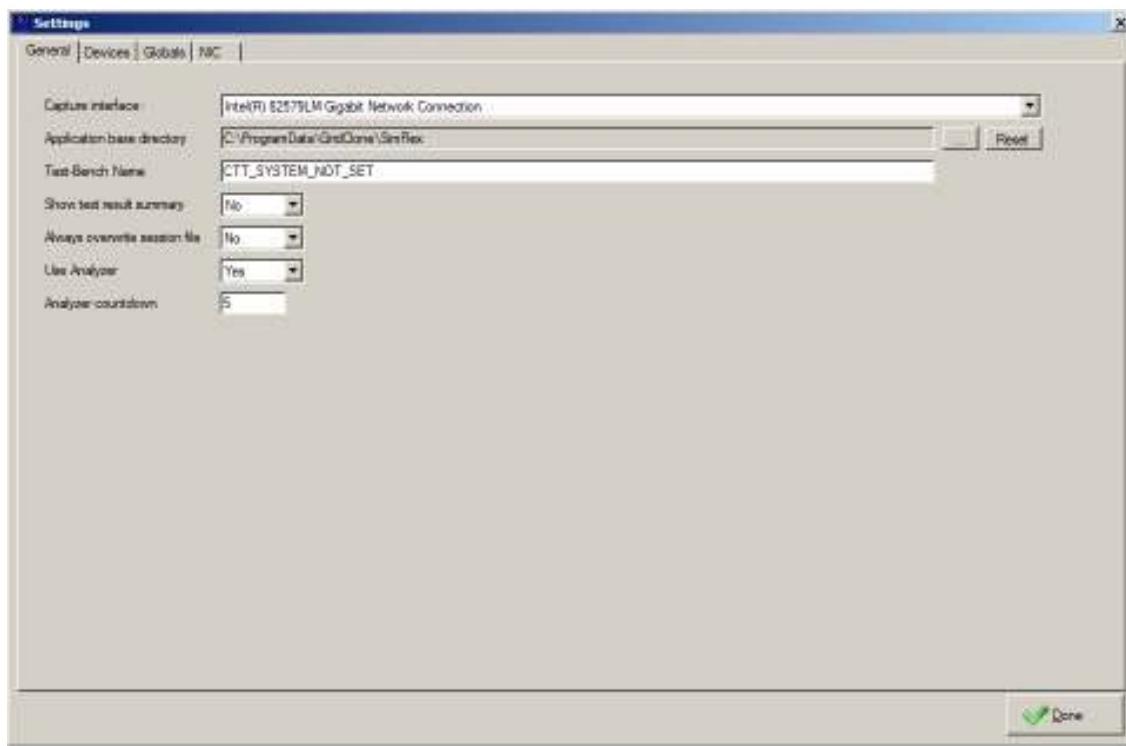


Figure 2.04

#### Capture interface

Select the Network Interface Card that will capture the network traffic during the execution of test cases

#### Application base directory

Select the directory that will be the base for the storage of all program related and created files. This includes amongst others the network capture files, the log files, the defined DUTs and all session related files.

<b>Test-Bench name</b>	Specify the name of the test bench. The test bench is the system on which the SimFlex™ Client Simulator has been installed on.
<b>Show test result summary</b>	Select <b>Yes</b> when a summary of the tests needs to be shown on the screen after a test run.
<b>Always overwrite session file</b>	Select <b>Yes</b> when the session file can be overwritten. This setting is only applicable when running test cases during a session.
<b>Use Analyzer</b>	Select <b>Yes</b> when <i>SimFlex™ Protocol Analyzer</i> actually is to be run when Analyze() is called in the script.
<b>Analyzer countdown</b>	Set the time the form that displays the issues found by <i>SimFlex™ Protocol Analyzer</i> has to be shown (in seconds).

## 2.4 Detailed information of Settings dialog, Devices tab

In the Settings dialog, Devices tab, you can define the Devices Under Test (DUTs). On the left side of the dialog a list of defined DUT is shown. Selecting a DUT shows the settings in the *Details* part of the dialog. You can add new devices by pressing the [ + Add ] button and remove devices by pressing the [ - Remove ] button. In the *Details* part of the dialog you can specify the **Device name**, **Vendor Name** and the **IP address** of the DUT. You can also enter text to describe the version of the DUT (e.g. the HW version, the SW version and the firmware version of the DUT). This text field is free-format. In the **Comments** field you can enter any information you may find important for the DUT.

The [ Advanced ] button will reveal fields that might be important during the association of the SimFlex™ Client Simulator with the DUT. By default the pre-configured values will do for most DUTs. Please refer to the PICS document of the manufacturer of the DUT for information regarding the settings. Changing these settings might cause the SimFlex™ Client Simulator not being able to properly connect to a DUT.

The following figure shows the settings that can be configured in the Devices tab.

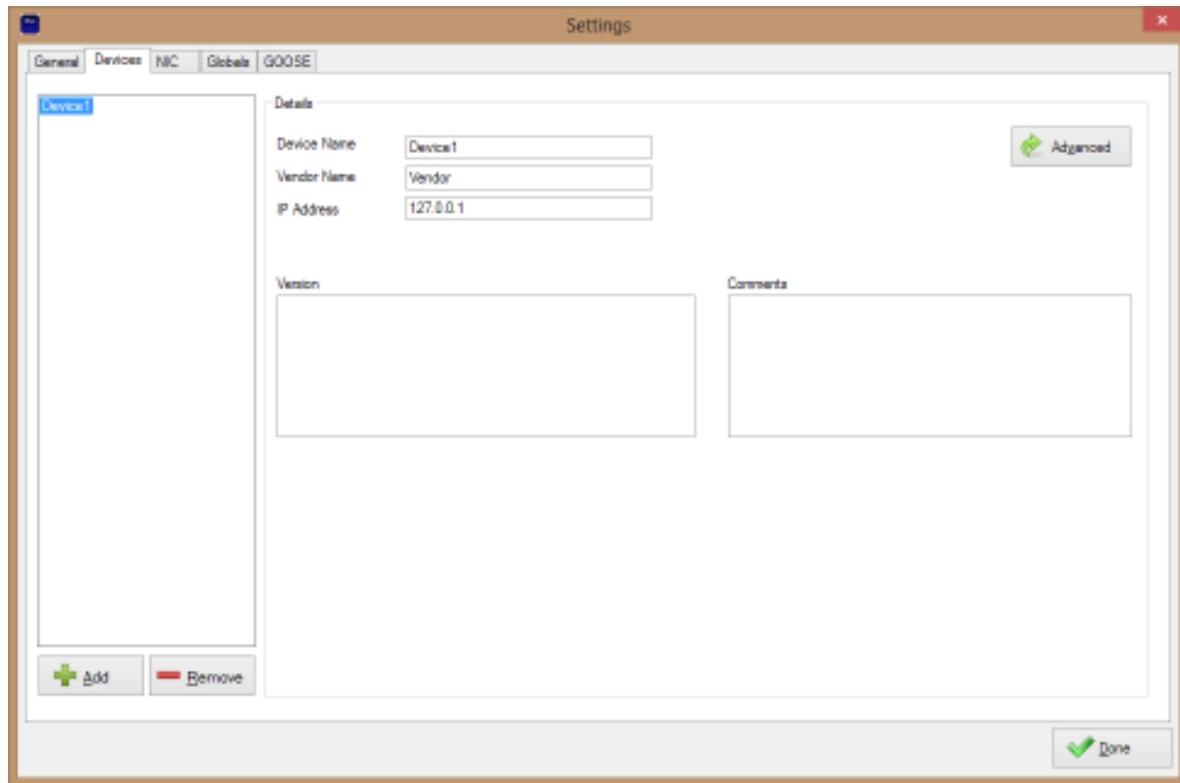
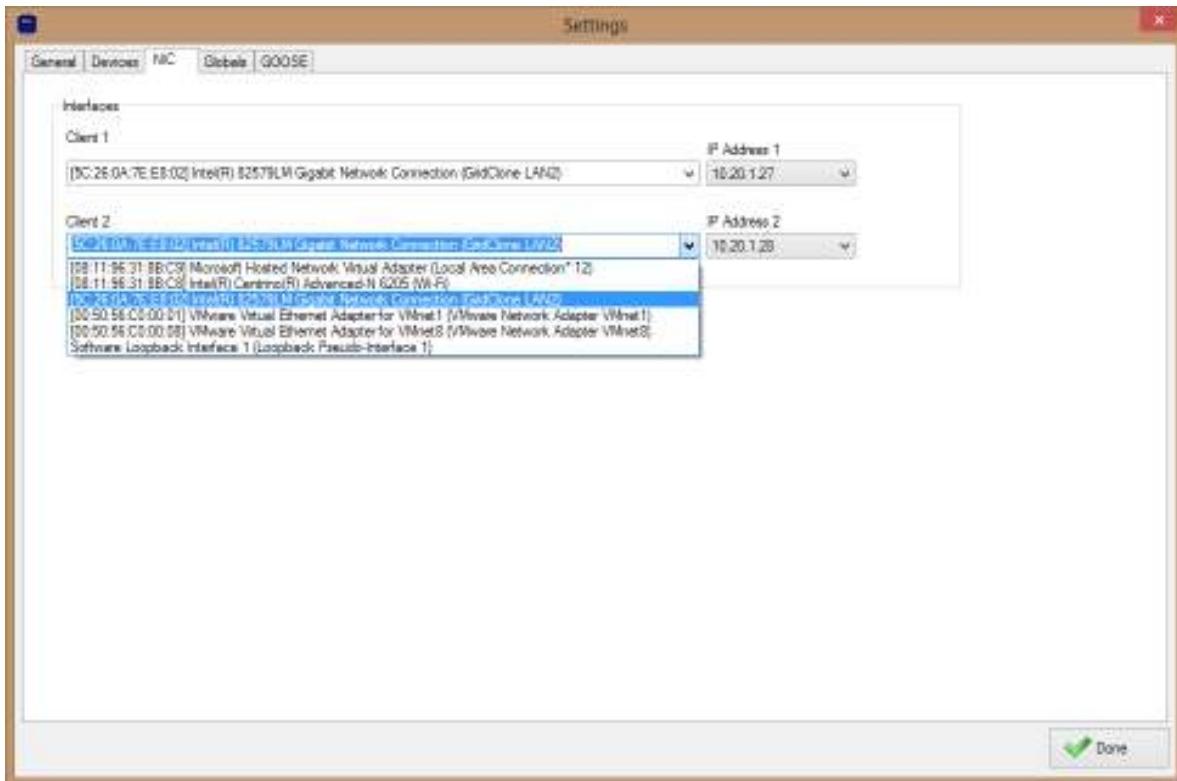


Figure 2.05

## 2.5 Detailed information of Settings dialog, NIC tab

In the NIC tab of the Settings window, the user must select the NIC interfaces for both clients, and one of the IP addresses that are available for each Network Interface Card (NIC) interface. The best configuration is when the PC has two independent Network Interface Cards, each with a different IP address. If your PC only has one NIC, please select that NIC for both Client 1 and Client 2, but select different IP addresses for both clients (both in the same IP address range as the DUT). If your PC only has one NIC and you see only one IP address in the drop-down list, please check your Windows OS manual on how to specify a second IP address on the same NIC.

Note that the NIC addresses must be in the same sub-network as the DUT.



**Figure 2.06**

## 2.6 Detailed information of Settings dialog, Globals tab

The SimFlex™ Client Simulator is a generic IEC 61850 Client capable of running test scripts (test cases) against many IEDs. The test scripts are defined in such a way that they should not need alteration when executed. Therefore the SimFlex™ Client Simulator needs input from the test engineer that can be defined in the Global tab. Most of the settings to be defined will come from the manufacturer's PICS and PIXIT documents.

**NOTE:** In most cases the DUT (Device Under Test) will support a subset of the conformance blocks as defined in the UCA© detailed test procedures for server devices. In that case only the settings for the applicable conformance blocks (CBs) need to be specified.

The settings are grouped per conformance block. The settings that belong to a specific conformance block can be identified by the Description. The description of each setting starts with a tag that identifies the conformance block the setting belongs to. E.g., settings that belong to conformance block 1 (Basic Exchange) start with "CB01", settings that belong to conformance block 2+ (Data Set Definition) start with "CB02+", etc.

The following figure shows part the settings that can be configured in the Globals tab. (Note: This figure can show different data than the form shown on your screen)

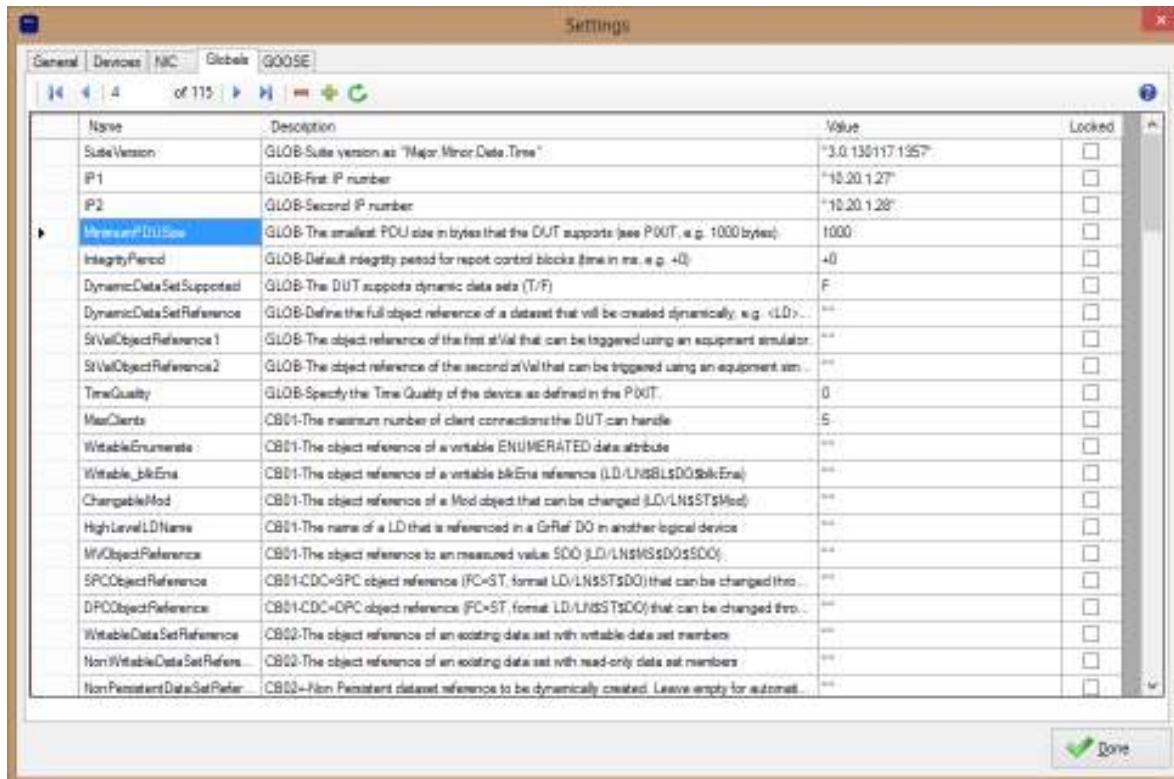


Figure 2.07

### 2.6.1 Program settings

The first three variables in the settings are informative and shall not be changed.

GROUP	NAME	DEFAULT VALUE	DESCRIPTION
# Program settings	SuiteVersion	"0.0.130415.1200"	GLOB-Suite version as "Major.Minor.Date.Time"
	IP1	"192.168.0.1"	GLOB-First IP number
	IP2	"192.168.0.2"	GLOB-Second IP number

### 2.6.2 Global test suite settings

The global test suite settings are not bound to test scripts in specific conformance block. They are used in test scripts throughout the whole test suite and need to be configured when applicable. The settings can be recognized by the “GLOB” prefix in the description column.

GROUP	NAME	VALUE	DESCRIPTION
Global test suite settings	MinimumPDUSize	1000	GLOB-The smallest PDU size in bytes that the DUT supports (see PIXIT, e.g. 1000 bytes).
	IntegrityPeriod	+0	GLOB-This is the default integrity period for report control blocks (time in ms e.g. +0). The value needs to be pre-fixed by a '+' sign.
	DynamicDataSetSupported	F	GLOB-Specify if the DUT supports the creation of dynamic data sets (T/F).
	DynamicDataSetReference	""	GLOB-Define the full object reference of a dataset name that will be created dynamically in case the creation of dynamic data sets is supported. Format:<LD>/<LN>\$<data set name>

			Leave empty ("") in case the creation of dynamic data sets is not supported by the DUT.
	StValObjectReference1	""	GLOB-Specify the object reference of a status value (stVal) that changes when an equipment simulator changes value. E.g. if the equipment simulator simulates the position of a breaker, this value will change accordingly.
	StValObjectReference2	""	GLOB- Specify the object reference of a second status value (stVal) that changes when an equipment simulator changes value. E.g. if the equipment simulator simulates the position of a breaker, this value will change accordingly. This object reference shall be different from the object reference specified in <i>StValObjectReference1</i> .
	TimeQuality	0	GLOB-Specify the Time Quality of the device as defined in the PIXIT. E.g. 10.

### 2.6.3 Settings for test cases in Conformance Block 1

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 1	MaxClients	5	CB01-Specify the maximum number of client connections that the DUT accepts (see PIXIT)
	WritableEnumerate	""	CB01- Specify the object reference of a writable ENUMERATED data attribute. Leave empty ("") when no writable ENUMERATED data attribute exists in the DUT.
	Writable_blkEna	""	CB01-The object reference of a writable blkEna reference (LD/LN\$BL\$DO\$blkEna)
	ChangableMod	""	CB01-The object reference of a writable blkEna reference (LD/LN\$BL\$DO\$blkEna)
	HighLevelLDName	""	CB01-The object reference of a Mod object that can be changed (LD/LN\$ST\$Mod)
	MVObjectReference	""	CB01-The name of a LD that is referenced in a GrRef DO in another logical device
	SPCObjectReference	""	CB01-The object reference to an measured value SDO (LD/LN\$MS\$DO\$SDO)
	DPCObjectReference	""	CB01-CDC=SPC object reference (FC=ST, format LD/LN\$ST\$DO) that can be changed through the EQUIPMENT SIMULATOR.

### 2.6.4 Settings for test cases in Conformance Block 2

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 2	WritableDataSetReference	""	CB02-Specify the object reference of an existing data set with writable data set members. Leave empty ("") when the DUT does not have a data set with writable members.
	NonWritableDataSetReference	""	CB02- Specify the object reference of an existing data set with read-only data set members. Leave empty ("") when the DUT does not have a data set with read-only members.

### 2.6.5 Settings for test cases in Conformance Block 2+

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 2+	NonPersistentDataSetReference	""	CB02+-Specify the object reference of a Non-Persistent data set that will be dynamically created. Leave empty ("") for automatic name creation.

	NonDeletableDataSetReference	""	CB02+-Specify a pre-configured (SCL), non-deletable data set that is <b>not</b> referenced in any report control block
	MaxNrOfDataSets	0	CB02+-Specify the maximum number of Persistent Datasets as defined by the manufacturer (PIXIT)
	MaxNrOfNonPersistentDataSets	0	CB02+-Specify the maximum number of Non-Persistent Datasets as defined by the manufacturer (PIXIT)
	MaxNrOfDatasetItems	0	CB02+-Specify the maximum number of Dataset Items as defined by the manufacturer (PIXIT)
	ExcludeFC	{"BR","RP","GO","CO"}	CB02+-Specify a list of Functional Constraints that shall not be used in the lookup of data set elements (object references) for the creation of a dynamic data set.

### 2.6.6 Settings for test cases in Conformance Block 3

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 3	SVVolatility	T	CB03-Specify whether or not substitution values are stored in volatile memory, (true=volatile)

### 2.6.7 Settings for test cases in Conformance Block 4

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 4	SGCB	""	CB04-The object reference of a Setting Group Control Block. Leave blank for automatic selection.

### 2.6.8 Settings for test cases in Conformance Block 5 & 6

GROUP	NAME	VALUE	DESCRIPTION
Variables for common report test cases (CB 5 and CB6)	ConfigurableDataSet	Dyn	CB05/6-ReportSettings: The DataSet attribute in the RCB is <b>Fix, Conf or Dyn</b>
	ConfigurableRptID	Dyn	CB05/6-ReportSettings: The RptID attribute in the RCB is <b>Fix, Conf or Dyn</b>
	ConfigurableOptFields	Dyn	CB05/6-ReportSettings: The OptFlds attribute in the RCB is <b>Fix, Conf or Dyn</b>
	ConfigurableBufTime	Dyn	CB05/6-ReportSettings: The BufTm attribute in the RCB is <b>Fix, Conf or Dyn</b>
	ConfigurableTrgOps	Dyn	CB05/6-ReportSettings: The TrgOps attribute in the RCB is <b>Fix, Conf or Dyn</b>
	ConfigurableIntgPd	Dyn	CB05/6-ReportSettings: The IntgPd attribute in the RCB is <b>Fix, Conf or Dyn</b>
Trigger Options bit-components	data_change	1	CB05/6-TrgOps: The DUT supports data-change (1=yes)
	quality_change	1	CB05/6-TrgOps: The DUT supports quality-change (1=yes)
	data_update	1	CB05/6-TrgOps: The DUT supports data-update (1=yes)
	integrity	1	CB05/6-TrgOps: The DUT supports integrity (1=yes)
	general_interrogation	1	CB05/6-TrgOps: The DUT supports general_interrogation (1=yes)
Optional Fields bit-components	sequence_number	1	CB05/6-OptFlds: The DUT supports optional field: sequence-number (1=yes)
	report_time_stamp	1	CB05/6-OptFlds: The DUT supports optional field: report-time-stamp (1=yes)
	reason_for_inclusion	1	CB05/6-OptFlds: The DUT supports optional field: reason-for-inclusion (1=yes)
	data_set_name	1	CB05/6-OptFlds: The DUT supports optional field: data-set-name (1=yes)
	data_reference	1	CB05/6-OptFlds: The DUT supports optional field: data-reference (1=yes)

	conf_revision	1	CB05/6-OptFlds: The DUT supports optional field: conf_revision (1=yes)
	BufferTimeNormal	+10000	CB05/6-The default value for BufTm in milliseconds (only change when needed)
	BufferTimeMargin	+1000	CB05/6-The acceptable margin for the inter-report time in milliseconds (needed for Rp7/Br7).
	AnalogueDataChange-Substitutes	F	CB05/6: A second analogue data change substitutes the first analogue data change when BufTm > 0 (check PIXIT for implemented behavior)
	AnalogueObjectReference1	""	CB05/6-Specify the object reference of the first analogue value that can be changed using an EQUIPMENT SIMULATOR.
	AnalogueObjectReference2	""	CB05/6-Specify the object reference of the second analogue value that can be changed using an EQUIPMENT SIMULATOR.
	PredefinedDataSet	""	CB05/6-Specify the object reference to a predefined data set (SCL), preferably with status attributes and analogue attributes that can be changed using an EQUIPMENT SIMULATOR. The data set elements shall be FCDAs (=data attributes), if available with stVal, q and t.
	PredefinedDataSetWithFCD	""	CB05/6-Specify the object reference to a predefined data set with FCDs (=data objects), with status values that can be changed using an EQUIPMENT SIMULATOR.
	VolatileRCBValues	F	CB05/6-With this variable the user can specify if the xRCB values ConfRev and DataSet are volatile, i.e. if they are restored to its original value after a reset of the DUT (see PIXIT).

### 2.6.9 Settings for test cases in Conformance Block 5

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 5	URCBReference	""	CB05-Specify the object reference to a URCB
	ConfiguredURCBReference	""	CB05-The object reference to a Configured URCB with max name length, max dataset length, and max rptID length (sRp14).
	LargeDataSetReference	""	CB05-Specify the object reference of a large data set that will be used for testing SEGMENTATION. Leave empty in case dynamic data sets are supported.
	PreAssignedURCBReference	""	CB05-Specify the object reference to a URCB that is pre-assigned to another client through SCL.

### 2.6.10 Settings for test cases in Conformance Block 6

GROUP	NAME	VALUE	DESCRIPTION
# Global Variables for CB 6	BRCBReference	""	CB06-Specify the object reference to a BRCB
	ConfiguredURCBReference	""	CB05-The object reference to a Configured URCB with max name length, max dataset length, and max rptID length (sRp14).
	buffer_overflow	1	CB06-OptFlds: The DUT supports optional field: buffer_overflow (1=yes)
	entryID	1	CB06-OptFlds: The DUT supports optional field: entryID (1=yes)
	PreAssignedBRCBReference	""	CB06- Specify the object reference to a BRCB that is pre-assigned to another client through SCL.

### 2.6.11 Settings for test cases in Conformance Block 9

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 9	GoCBReference	""	CB09-Specify the GoCB reference from the SCL file that is used by the DUT to publish GOOSE messages.
	LGOSReference	""	CB09-Specify the LGOS reference (LD/xxLGOSn) for the LGOS logical node that contains the subscription information for the GOOSE published by the simulator.
	ExpectedResultIsSuccess_GopN1	T	CB09-Specify the expected behavior of the device: 'T' if the GoCB can be enabled/disabled via SetGoCBValues (see PIXIT).
	GetGoCBValuesSupported	F	CB09-DUT supports GetGoCBValues (T/F).
	SetGoCBValuesSupported	F	CB09-DUT supports SetGoCBValues (T/F).
	GooseRepeatCount	+10	CB09-The number times a GOOSE messages sent by the Client simulator for GOOSE subscribe testing is repeated

### 2.6.12 Common Settings for test cases in Conformance Blocks 12

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 12	ValidOrCat	3	CB12-Specify a SUPPORTED orCat value for the DUT.
	InvalidOrCat	0	CB12-Specify an UNSUPPORTED orCat value for the DUT (E.g. 0).
	OperateTimeout	10000	CB12-Specify the Operate timeout of the DUT as defined in the PIXIT (time in milliseconds)
	sboTimeout	300000	CB12-Specify the SBO timeout value of the DUT as defined in the PIXIT (time in milliseconds)
	ParameterToChange	""	CB12-User set the reference of a parameter that will be manually changed between a Select and an Operate in sCtlI20 (see PIXIT)
	ChangeableControl	""	CB12-User set the path of a Control object with a writable ctIModel, for example "LD_CTRL/LLN0\$CO\$LEDRs"

### 2.6.13 Settings for test cases in Conformance Block 12a

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 12a	DOnsObject	""	CB12a-User set the path of a DOns object, for example "LD_CTRL/LLN0\$CO\$LEDRs"
	TADOnsObject1	""	CB12a-User set the path of a DOns object that supports TimeActivated control, for example "LD_CTRL/CSWI1\$CO\$Pos"
	TADOnsObject2	""	CB12a-User set the path of a second DOns object (TADOnsObject1 != TADOnsObject2) that supports TimeActivated control, for example "LD_CTRL/CSWI2\$CO\$Pos"
	TapPosDOnsObject	""	CB12a-User set the path of a Tap Position DOns object, for example "LD_CTRL/ATCC1\$CO\$TapPos"
	TapChgDOnsObject	""	CB12a-User set the path of a Tap Changer DOns object, for example "LD_CTRL/ATCC1\$CO\$TapChg"
	APCDOnsObject	""	CB12a-User set the path of a APC DOns object, for example "LD/LN\$CO\$ColPos"
	ExpectedResult_Ctl10a	"success"	CB12a-Select the expected behavior of the device as specified in PIXIT: success/no-addcause/position-reached/time-limit
	ExpectedResult_Ctl14a	"success"	CB12a-Select the expected behavior of the device as specified in PIXIT: success/no-addcause/command-already-execution

### 2.6.14 Settings for test cases in Conformance Block 12b

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 12b	SBOsObject	""	CB12b-User set the path of a SBOs object, for example "LD_CTRL/CSWI1\$CO\$Pos".
	SBOsObjects	{"","","","",""}	CB12b-User set the list of paths to some SBOs objects, for example: {"LD_CTRL/CSWI1\$CO\$Pos","LD_CTRL/CSWI2\$CO\$Pos"}
	TASBOnsObject1	""	CB12b-User set the path of a SBOs object that supports TimeActivated control, for example "LD_CTRL/CSWI1\$CO\$Pos"
	TASBOnsObject2	""	CB12b-User set the path of a second SBOs object (TASBOnsObject2 != TASBOnsObject1) that supports TimeActivated control, for example "LD_CTRL/CSWI2\$CO\$Pos"
	TapPosSBOsObject	""	CB12b-User set the path of a Tap Position SBOs object, for example "LD_CTRL/ATCC1\$CO\$TapPos"
	TapChgSBOsObject	""	CB12b-User set the path of a Tap Changer SBOs object, for example "LD_CTRL/ATCC1\$CO\$TapChg"
	APCSBOnsObject	""	CB12b-User set the path of a APC SBOs object, for example "LD/LN\$CO\$ColPos"
	ExpectedResult_Ctl10b	"success"	CB12b-Select the expected behavior of the device as specified in PIXIT: success/no-addcause/position-reached/time-limit

### 2.6.15 Settings for test cases in Conformance Block 12c

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 12c	DOesObject	""	CB12c-User set the path of a DOes object, for example "LD_CTRL/LLNO\$CO\$LEDs"
	TADOesObject1	""	CB12c-User set the path of a DOes object that supports TimeActivated control, for example "LD_CTRL/CSWI1\$CO\$Pos"
	TADOesObject2	""	CB12c-User set the path of a second DOes object (TADOesObject1 != TADOesObject2) that supports TimeActivated control, for example "LD_CTRL/CSWI2\$CO\$Pos"
	TapPosDOesObject	""	CB12c-User set the path of a Tap Position DOes object, for example "LD_CTRL/ATCC1\$CO\$TapPos"
	TapChgDOesObject	""	CB12c-User set the path of a Tap Changer DOes object, for example "LD_CTRL/ATCC1\$CO\$TapChg"
	APCDOesObject	""	CB12c-User set the path of a APC DOes object, for example "LD/LN\$CO\$ColPos"
	ExpectedResult_Ctl10c	"success"	CB12c-Select the expected behavior of the device as specified in PIXIT: success/position-reached/time-limit
	ExpectedResult_Ctl14c	"success"	CB12c-Select the expected behavior of the device as specified in PIXIT: success/command-already-execution

### 2.6.16 Settings for test cases in Conformance Block 12d

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 12d	SBOesObject	""	CB12d-User set the path of a SBOes object, for example "LD_CTRL/CSWI1\$CO\$Pos"
	SBOesObjects	{"","","","",""}	CB12d-User set the list of paths to the SBOes objects, for example {"LD_CTRL/CSWI1\$CO\$Pos","LD_CTRL/CSWI2\$CO\$Pos"} .

	TASBOesObject1	""	CB12d-User set the path of a SBOes object that supports TimeActivated control, for example "LD_CTRL/CSWI1\$CO\$Pos"
	TASBOesObject2	""	CB12d-User set the path of a second SBOes object (TASBOesObject2 != TASBOesObject1) that supports TimeActivated control, for example "LD_CTRL/CSWI2\$CO\$Pos"
	TapPosSBOesObject	""	CB12d-User set the path of a Tap Position SBOes object, for example "LD_CTRL/ATCC1\$CO\$TapPos"
	TapChgSBOesObject	""	CB12d-User set the path of a Tap Changer SBOes object, for example "LD_CTRL/ATCC1\$CO\$TapPos"
	APCSBOesObject	""	CB12d-User set the path of an APC SBOes object, for example "LD_LN\$CO\$ColPos"
	ExpectedResult_Ctl10d	"success"	CB12d-Select the expected behavior of the device as specified in PIXIT: success/select-position-reached/select-time-limit/position-reached/time-limit

### 2.6.17 Settings for test cases in Conformance Block 13

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 13	EventDrivenUTCTime	""	CB13-Specify an object reference to a UTC timestamp (e.g., data attribute 't') that changes when the data object that the time stamp belongs to (e.g. data object 'Pos') is triggered by an event created by the EQUIPMENT SIMULATOR (E.g. "LD_CTRL/CSWI1\$ST\$Pos\$t")
	SyncLosingPeriod	0	CB13-Specify the period the device will lose time synchronisation with the time server as specified in PIXIT (time in milliseconds).
	DisturbanceRecordDir	""	CB13-Specify the directory in the DUT where disturbance records are stored.

### 2.6.18 Settings for test cases in Conformance Block 14

GROUP	NAME	VALUE	DESCRIPTION
Variables for CB 14	LargeFileReference	""	CB14-Select the path to a large file for Ft4, the bigger the file the more likely the test will be successful. Leave it empty for automatic file selection.
	LocalFileDir	""	CB14-Specify a directory containing files that can be uploaded to the DUT.
	RemoteFileDir	""	CB14-Specify a directory that can be written in the DUT.
	maxFileSizeBytes	1048576	CB14-Specify the maximum supported file size by the DUT in bytes.
	ExpectedResultIsSuccess_Ft4	T	CB14-Select the expected behavior of the device as specified in PIXIT

## 3 Use case: Starting and stopping SimFlex CS

### 3.1 Verify a connection with a DUT

Make sure that both IED and the PC that is running the SimFlex CS are connected to the same network and that they are in the same IP address range.

Example:

#### IED

IP address = 192.168.0.100, mask = 255.255.255.0

#### SimFlex Client Simulator

IP address = 192.168.0.90, mask 255.255.255.0

Choose the right interface that connects the client (PC used to run SimFlex CS) and the server/device.

**NOTE:** Please check your Windows OS manual on how to change adapt the IP address and network mask.

### 3.2 Start SimFlex CS

Open the SimFlex Client Simulator as you would normally open any Windows application. You may find a link under the Start menu in the GridClone folder.

Another way to start the program is to double-click the SimFlex Client Simulator icon on the desktop (available when this option has selected during the installation of the software).

### 3.3 Configure the right network adapter in SimFlex CS

Please check chapter 2.5 *Detailed information of Settings dialog, NIC tab* for more information on how to configure the network interface cards in the SimFlex™ Client Simulator.

### 3.4 Select, Edit or Add a DUT to SimFlex CS

Use the menu item **Devices** to **Edit**, **Add** or **Select** new devices.



Figure 3.01

You may also use the button **Devices**.



Figure 3.02

### 3.5 Build the data model

Hover over the icon "Double click to start creation of data model". The user can choose between build normal Data Model or Data Model with additional Read/Write information, which will be more useful but take longer time to be generated.

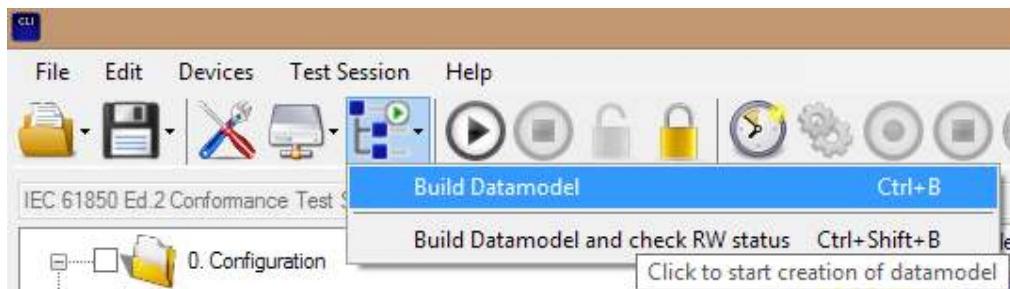


Figure 3.03

## 4 Use case: Run a single test

This section explains what a test suite is, how to load a test suite into SimFlex CS, how to run one or more tests and how to evaluate the results of the run tests.

First of all, what is a test suite? A test suite is a compilation of several test scripts grouped together in one file. The extension of test suite files is ".suite". In addition to the test scripts, a test suite also contains the global variables and some metadata. Internally it is a XML file but it is not intended to be edited by the end user, instead the SimFlex CS should be used for any editing of a test suite.

### 4.1 Open a test suite

In order to load a test suite from the SimFlex CS the user has several options:

- 1) Click in the menu bar under: **File → Open → Open Test Suite**, then select a test suite in the *Open Test Suite* dialog window.

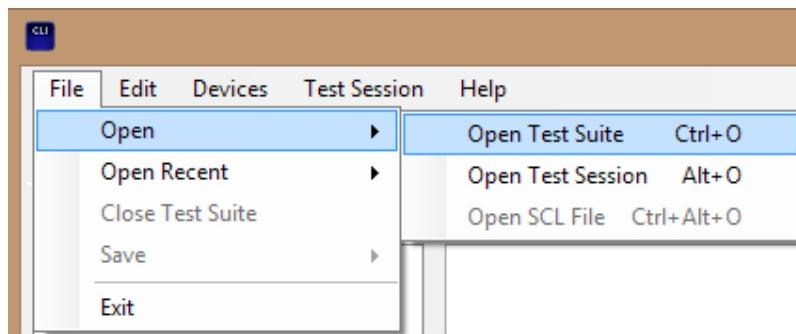


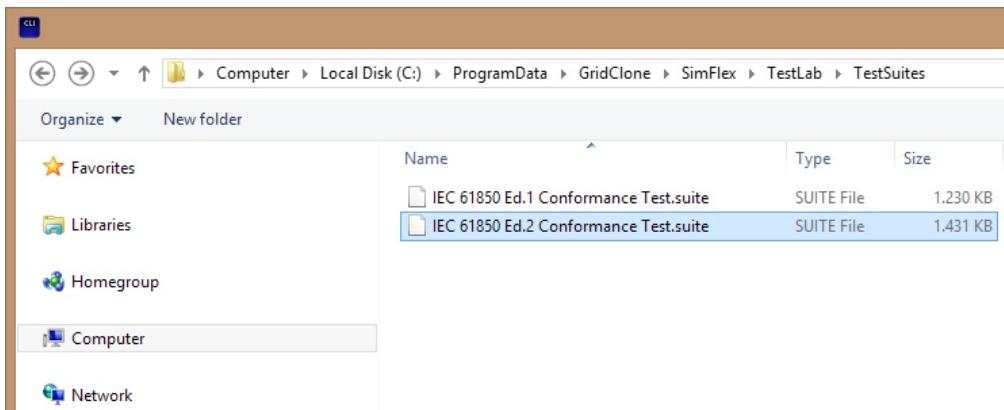
Figure 4.1

- 2) Click in the tool bar button: **Open → Open Test Suite**, then select a test suite in the *Open Test Suite* dialog window.



**Figure 4.2**

- 3) Press the shortcut **CTRL + O** and then select a test suite in the *Open Test Suite* dialog window.

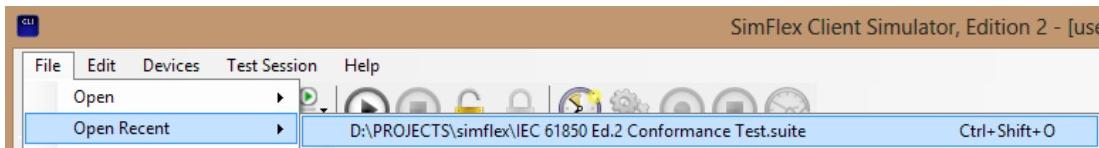


**Figure 4.3**

Note: The Open Test Suite dialog window will be opened in the Test suites folder (by default in “C:\ProgramData\GridClone\SimFlex\TestLab\TestSuites”). If the user changes the “*Application base folder*” as explained in Section 2 the *Test Suites* folder will change accordingly.

**NOTE:** Initially the folder “C:\ProgramData\GridClone\SimFlex\TestLab\TestSuites” might not contain the IEC 61850 Edition 2 test suite. In that case the test suite is located in the installation folder of the SimFlex CS, e.g. “C:\Program Files (x86)\GridClone\SimFlex Client Simulator\testsuites” (32bit version of Windows) or “C:\Program Files (x86)\GridClone\SimFlex Client Simulator\testsuites” (64bits version of Windows).

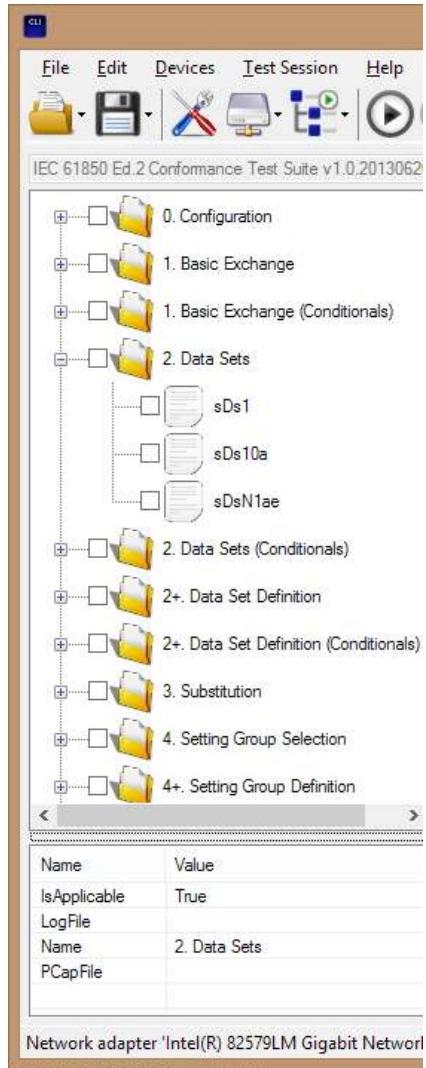
It is also possible to load a recently opened test suite selecting the wanted test suite from the menu bar under: **File → Open Recent** and selecting a “.suite” file.



**Figure 4.4**

## 4.2 Test suite tree panel and script tabs

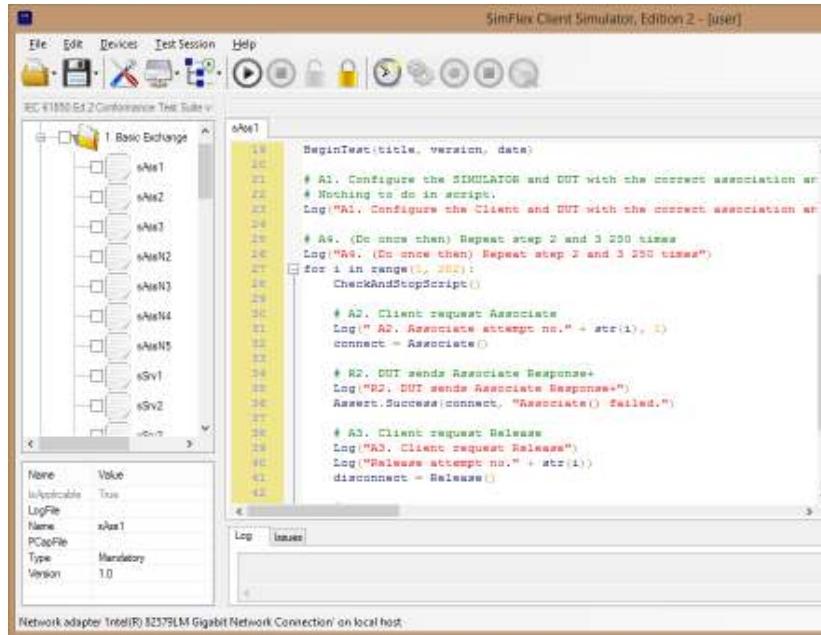
Providing that the file is a valid suite file, it will be shown in the left panel of the SimFlex CS.



**Figure 4.5**

The top label (1) shows the name and version of the loaded test suite. The central part (2) shows a tree model of the test suite with conformance blocks as nodes (3) and test scripts as leaves (4). Finally, the lower part of the panel (5) shows details of the selected (if any) test script or conformance block.

The content of the test scripts can be shown by double clicking in a test script. A tab will be opened in the central panel of the SimFlex CS as shown in the next figure:



**Figure 4.6**

### 4.3 Select and run test scripts

Select at least one test script in at least one conformance block and then click the **Run** button.



**Figure 4.7**

If more than one test script is selected in the active conformance block and/or more than one conformance block is selected with active scripts in it, all of the active scripts will be run sequentially from the top.

The user might stop the tests pressing the **Stop** button that is located next to the **Run** button. The current running test will be aborted and none of the remaining tests will be run.

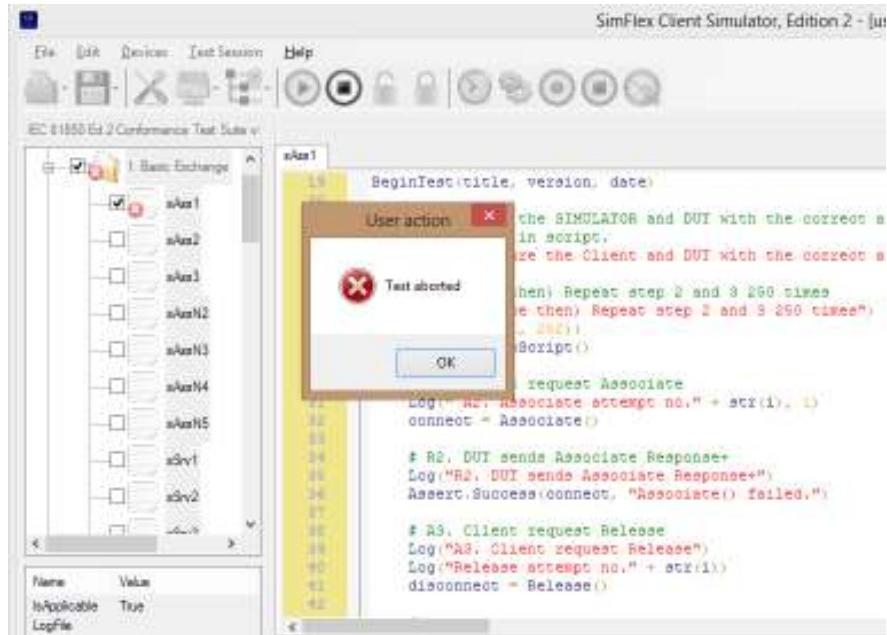


Figure 4.8

While a test is running, some log messages will be shown in the *Log* panel. This panel shows useful information of every step in the script and it is invaluable to know why a test is failing.

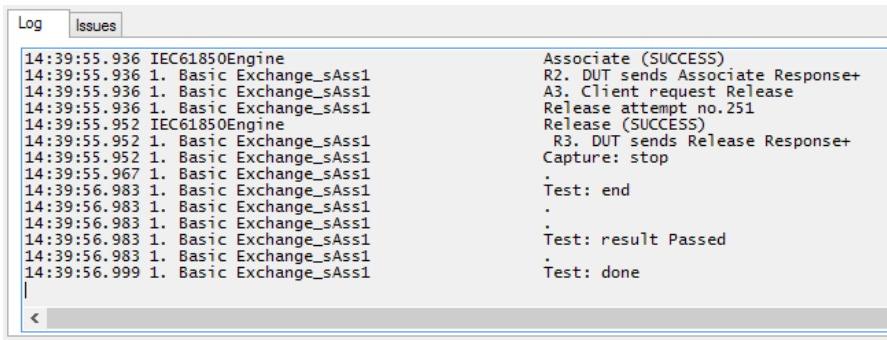
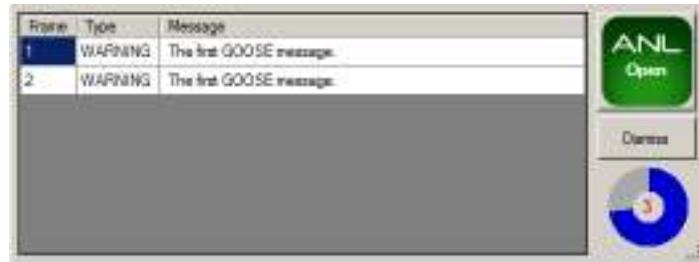


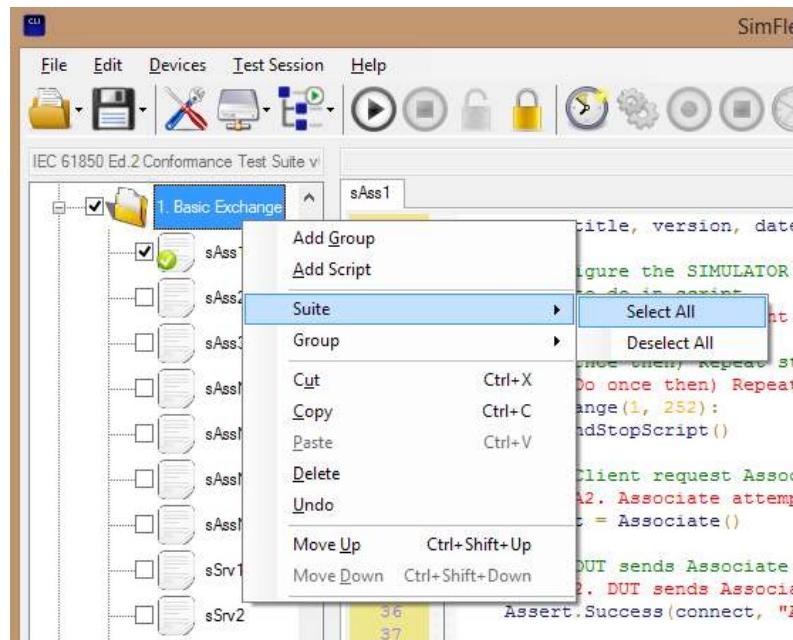
Figure 4.9

If *SimFlex™ Protocol Analyzer* is used and issues are found in the capture file. These issues will be shown in a form. The form will automatically close after the countdown time that is set. Closing can be forced by clicking *Dismiss*. Countdown can be stopped by clicking in the table or in the countdown clock. *SimFlex™ Protocol Analyzer* can be started using the *ANL Open* button or by double click in the row. A new session of *SimFlex™ Protocol Analyzer* will be started, the capture file will be opened, *Analyze* will be run and, if opened by double click, the frame that is clicked on will be activated.



**Figure 4.10**

It is possible to select and deselect all scripts in the suite by right-clicking anywhere in the suite and selecting **Suite → Select All / Deselect All**.



**Figure 4.11**

It is also possible to select and deselect all scripts in a group by right-clicking in a script or group and then selecting **Group → Select All / Deselect All**.

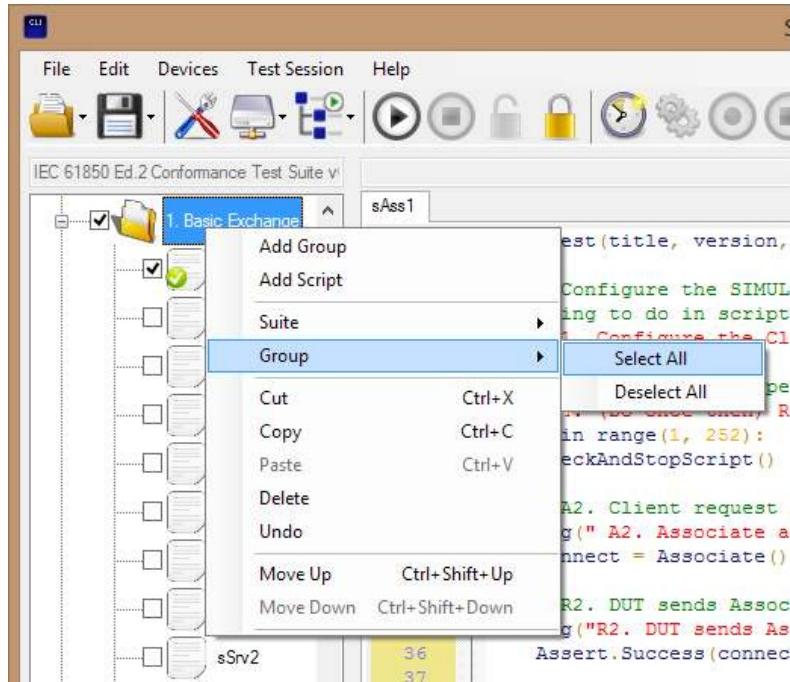


Figure 4.12

#### 4.4 Evaluate test results

Each script may have a status indication in the left panel as shown in the following figure:

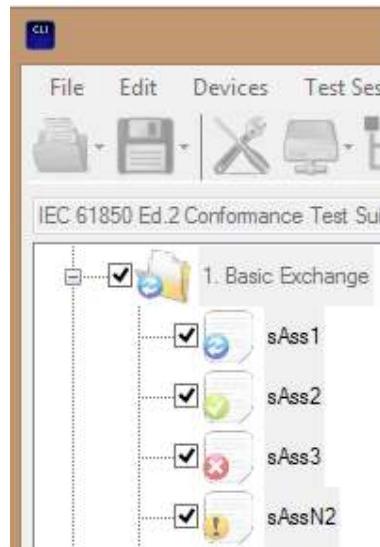


Figure 4.13

The currently running test will have the *Running* status icon (blue). All run tests will have either the *Pass* (green) or the *Fail* (red) status icon. The *Inconclusive* (yellow) status can only be set manually.

The user can override the result status of any test (e.g. if the test engineer sees some non-conformant behavior in the device) by right clicking at the test script and clicking on ***Change Status →(Passed | Failed | Inconclusive | Unknown)***.

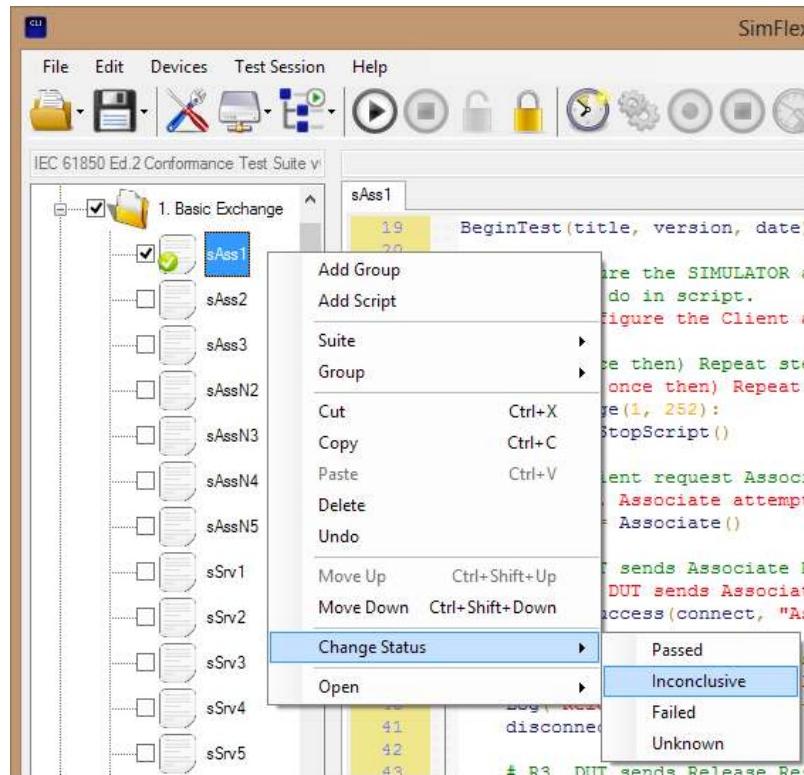


Figure 4.14

## 4.5 Open and examine a log file

If a test that has already been run is selected, the *Log* panel will show the log file for that specific test. The log file contains the same information as shown while running the test, and also shows a header and footer with useful information.

The log file can also be opened with an external editor (e.g. Notepad). Right click on the desired test and then ***Open* → *Log file***.

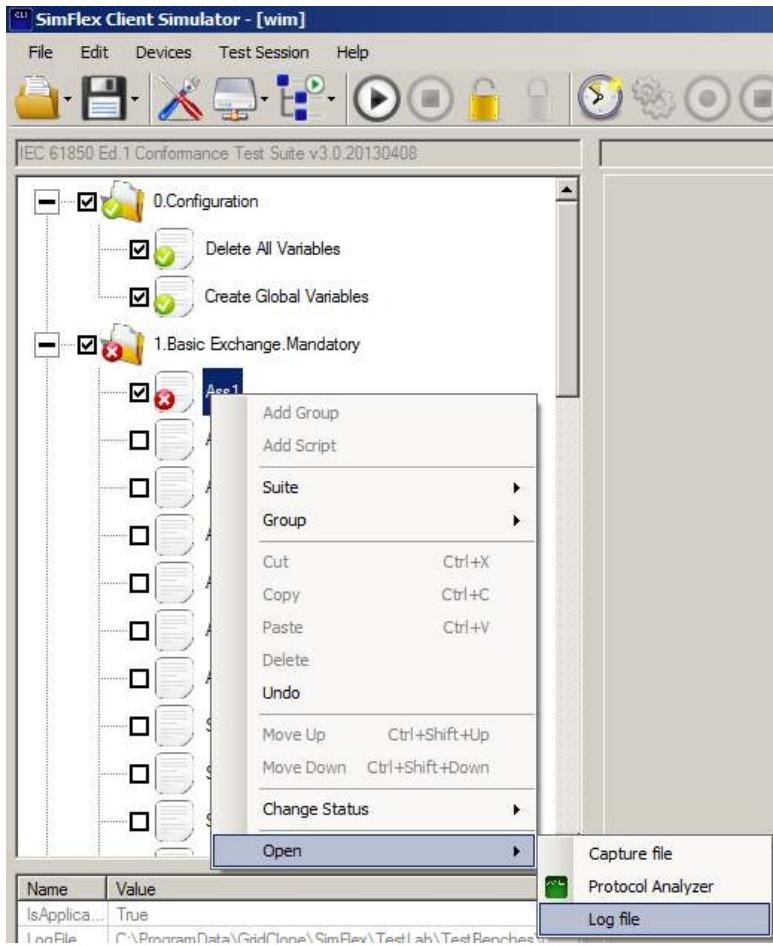


Figure 4.15

Windows will choose the default application for text files and open the selected log file. You can change the default application to use in the Windows settings.

#### 4.6 Open and examine a capture file

In addition to log files, the SimFlex CS also stores a capture file of all the Ethernet packets in the network. That capture file can be opened and examined by right clicking in an already run test and then **Open → Capture file**. Windows will choose the default application for ".pcap" files. You can change the default application to use in the Windows settings.

**NOTE:** To view the capture file, a third-party tool capable of showing the contents of the ".pcap" file format needs to be installed. A well-known example is **Wireshark** (see <http://www.wireshark.org/>).

If installed, the capture file can also be opened in the *SimFlex™ Protocol Analyzer*. If right click + then **Open → Protocol Analyzer** is selected, a new session of *SimFlex™ Protocol Analyzer* will be started, the capture file will be opened and *Analyze* will be run.

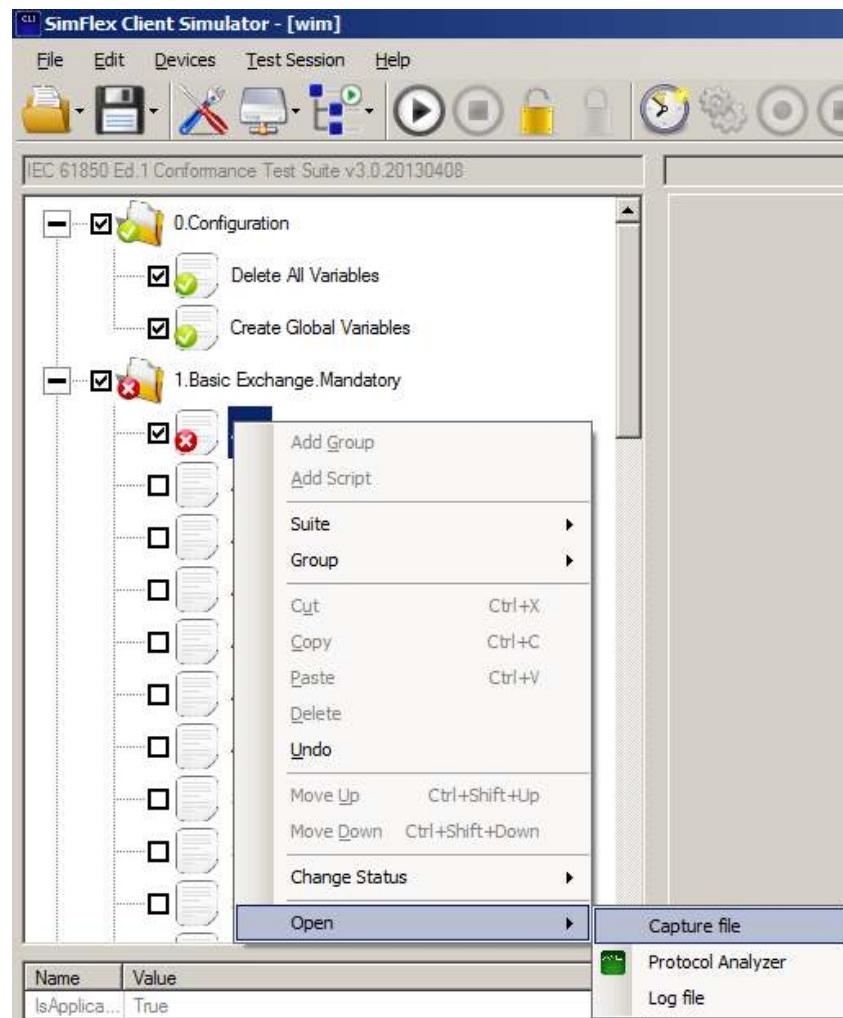


Figure 4.16

#### 4.7 Save and close a test suite

After modifying the selection of scripts, the Global Settings (as explained in Section 2) or the contents of a script (as will be explained in Section 6) the user may want to save manually the test suite. In order to do so there are two methods:

- 1) Click in the menu bar under: **File → Save → Save Test Suite.**



Figure 4.17

- 2) Click in the tool bar button: **Save → Save Test Suite.**

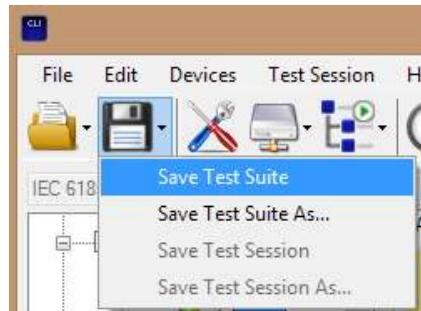


Figure 4.18

- 3) Press the shortcut **CTRL + S**.

Additionally, test suites will be automatically saved in the following cases: before a test is run, before the suite is closed (with a confirmation warning) and before the application is closed (with a confirmation warning).

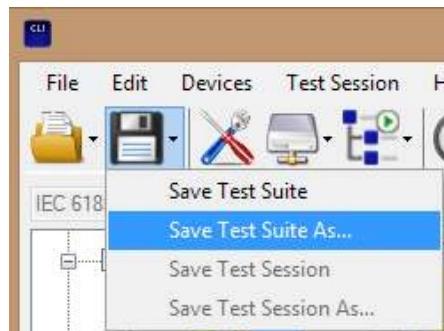
The user may want to save the test suite as a different file. To do so:

- 1) Click in the menu bar under: **File → Save → Save Test Suite As...**



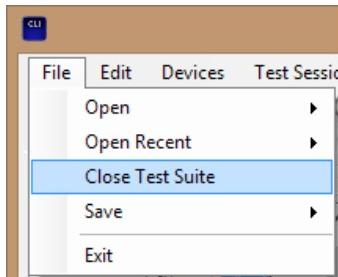
Figure 4.19

- 2) Click in the tool bar button: **Save → Save Test Suite As...**



**Figure 4.20**

The user can close a test suite manually by clicking in the menu bar under: **File → Close Test Suite**.



**Figure 4.21**

Other actions that will close the test suite are: opening another test suite, opening or creating a test session and closing the application.

## 5 Use case: Run a session

This section explains what a test session is, how to create or load a test session into SimFlex CS, how to run one or more tests in a session and how to evaluate the results of the run tests.

First of all, what is a test session? A test session is a compilation of a test suite and session information like DUT, test lab, vendor and settings grouped together in one file. The extension of test session files is ".tsf". Internally it is a XML file but it is not intended to be edited by the end user, instead the SimFlex CS should be used for any editing of a test session.

Note that test sessions created with SimFlex™ CS Edition 1 cannot be loaded with SimFlex™ CS Edition 2 due to differences in model and test structures.

### 5.1 Create a test session

A test session can be created in two different ways:

- 1) Click in the menu bar under: **Test Session → Create**.



**Figure 5.01**

- 2) Click in the tool bar button: **Create a new test session**.

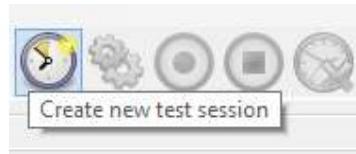


Figure 5.02

The next step is to select a DUT for the session as shown in the next figure:

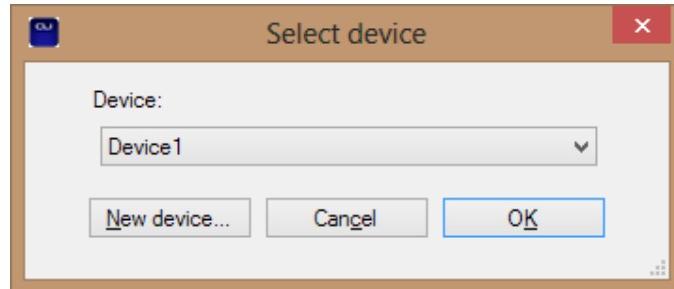


Figure 5.03

Also a test suite must be selected as shown in the next figure:

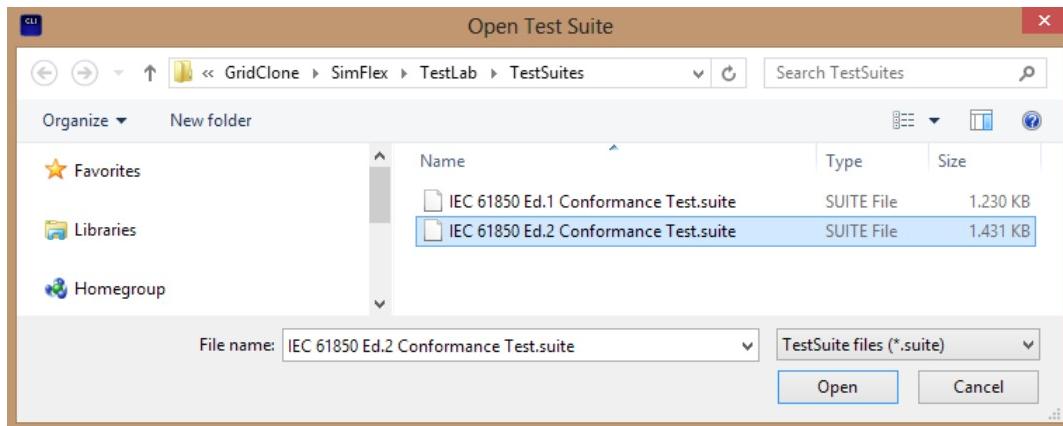
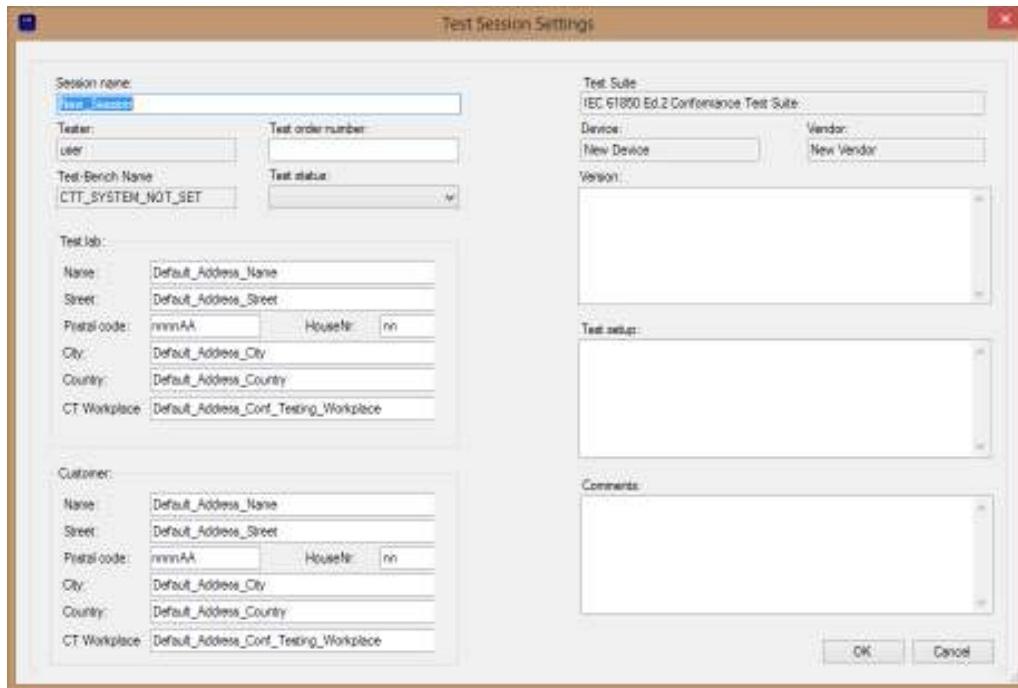


Figure 5.04

Finally, the user must fill in the session information in the session settings form. The several fields to fill in will be explained in a following section. For now, the session name and test session number are the only mandatory fields to be filled in.

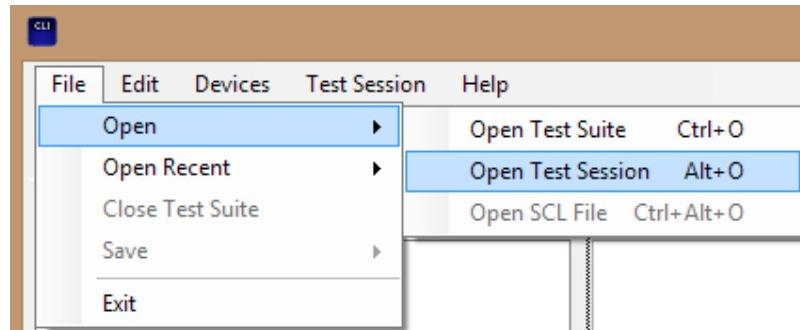


**Figure 5.05**

## 5.2 Open a test session

In order to open a test session that was previously saved, the user has several options:

- 1) Click in the menu bar under: **File → Open → Open Test Session**, then select a test session in the *Open Test Session* dialog window.



**Figure 5.06**

- 2) Click in the menu bar under: **Test Session → Open**, then select a test session in the *Open Test Session* dialog window.

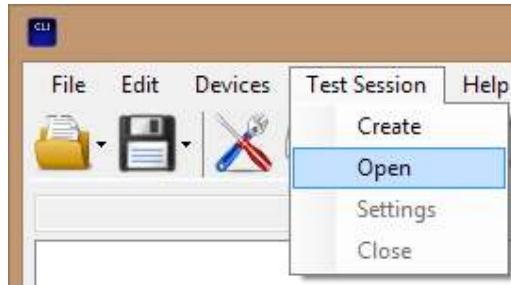


Figure 5.07

- 3) Click in the tool bar button: **Open → Open Test Session** and then select a test session in the *Open Test Session* dialog window.



Figure 5.08

- 4) Press the shortcut **ALT + O** and then select a test session in the *Open Test Session* dialog window.

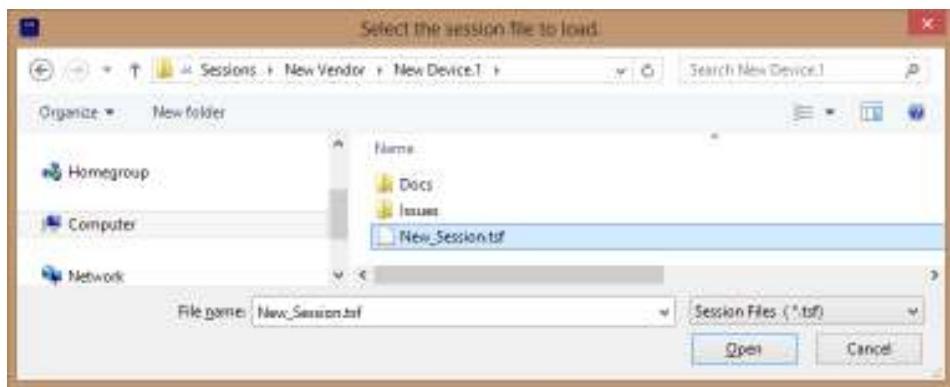


Figure 5.09

Note: The Open Test Session dialog window will be opened in the Sessions folder (by default in "C:\ProgramData\GridClone\SimFlex\Sessions"). If the user changes the "*Application base folder*" as explained in Section 2 the Sessions folder will change accordingly.

It is also possible to load a recently opened test session selecting the wanted test session from the menu bar under: **File → Open Recent** and selecting a ".tsf" file.

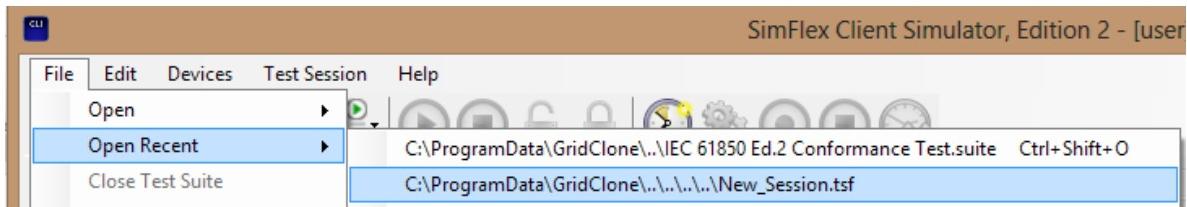


Figure 5.10

### 5.3 Run tests in ‘non-recording’ mode

Tests can be run as explained in Section 4. One or more test scripts and conformance blocks can be selected and run. The status icons will change accordingly and the log files and capture files can be inspected as explained in Section 4.

Evaluating the test results and opening log and capture files can be accomplished as explained in Section 4.

Running scripts in this mode is intended for fine-tuning or adjusting the settings of the DUT, suite and/or session. After that, the test engineer may want to go on ‘record’ mode.

### 5.4 Run tests in ‘recording’ mode

The user can press the *Record* button to go on ‘recording’ mode. The *Record* button will be disabled and a new *Stop* button will be enabled. To stop recording the session press the *Stop* button.



Figure 5.11

Another side effect of going into ‘recording’ mode is that all status icons of the scripts run in ‘non-recording’ mode will be cleared. Some other scripts may be set to other statuses depending on previous runs in ‘recording’ mode. In short, when a test is run in ‘recording’ mode its status, log file and capture file information will be stored in the session file. Thus, that information can be retrieved at any moment by the test engineer. Note that only the last run of the script will be stored, the user must take care of not overwriting the results by re-running the test.

To run a test, like in the other cases, select the appropriate tests and press the Run button. A warning window may appear before and after running the tests to confirm the overwriting of the test session file. The first confirmation allows overwriting any changes in the test suite and the second one allows overwriting of the test results.

Evaluating the test results and opening log and capture files can be accomplished as explained in Section 4. If for any reason, the test engineer desires to overwrite a test status, it can be done as explained in Section 4. This value will be stored in the session file and will be available in future runs of the session.

### 5.5 Save and close test sessions

After modifying a session, it can be manually saved by three methods:

- 1) Click in the menu bar under: **File → Save → Save Test Session.**



Figure 5.12

- 2) Click in the tool bar button: **Save → Save Test Session.**

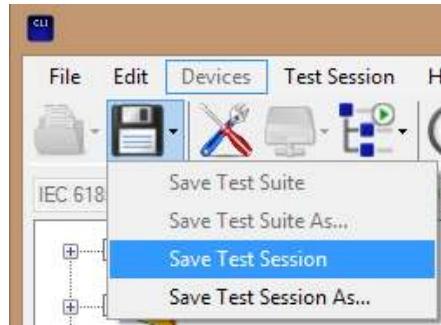


Figure 5.13

- 3) Press the shortcut **CTRL + ALT + S**.

Additionally, test sessions will be automatically saved (with a confirmation warning) in the following cases: before and after a test is run, before the session is closed and before the application is closed.

The user can close a test session manually by:

- 1) Clicking in the menu bar under: **Test Session → Close.**



Figure 5.14

- 2) Clicking in the tool bar button: **Close Test Session.**



**Figure 5.15**

## 5.6 Modify and copy test sessions

In addition to changes in the session's tests and results, the user can modify the session settings by:

- 1) Clicking in the menu bar under: **Test Session → Settings**.



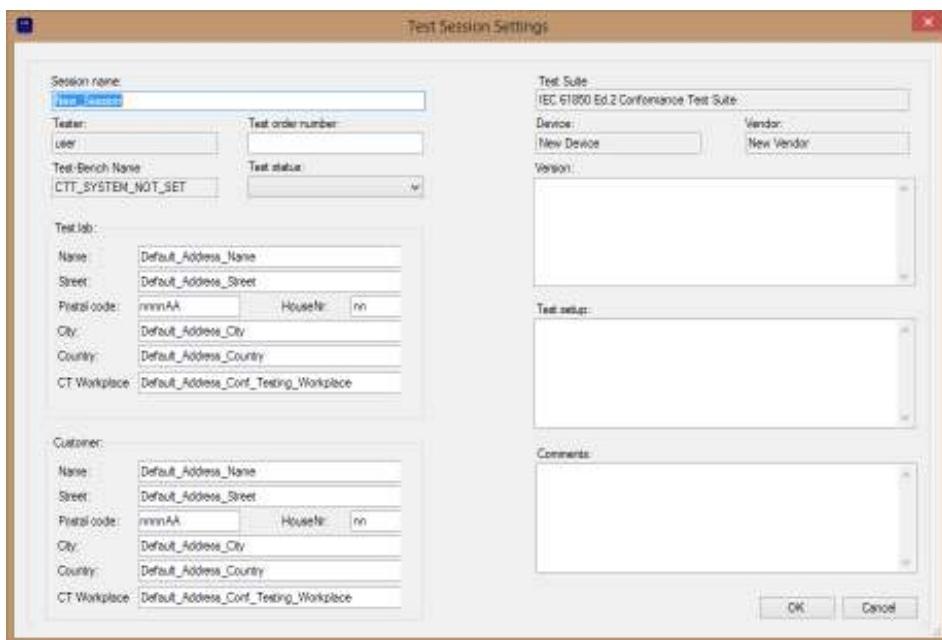
**Figure 5.16**

- 2) Clicking in the tool bar button: **Test Session Settings**.



**Figure 5.17**

The session settings window will appear as shown in next figure:



**Figure 5.18**

The following fields can be changed freely as they are only used as information containers: ‘Test status’, ‘Test lab’ address, ‘Customer’ address, DUT ‘Version’, ‘Test setup’ and ‘Comments’. After modifying any of these fields the user can press the ‘**Ok**’ button to store the settings or the ‘**Cancel**’ button to discard the changes. After pressing ‘**Ok**’ a confirmation window will appear.

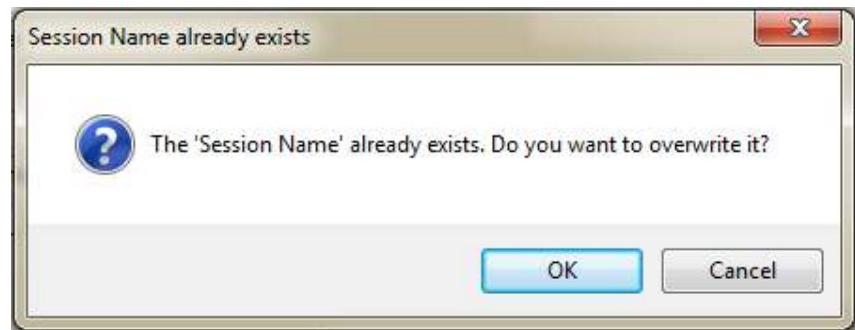


Figure 5.19

Special care must be taken with the ‘Session name’ and ‘Test order number’ fields because these two have side effects. Namely, changing the session name will also change the file name in which the session is saved. Changing the ‘Test order number’ will prompt a warning window. Pressing ‘**Yes**’ will copy the current session (with the exception of the test results) to a new folder location. Pressing ‘**No**’ will rename the current session.

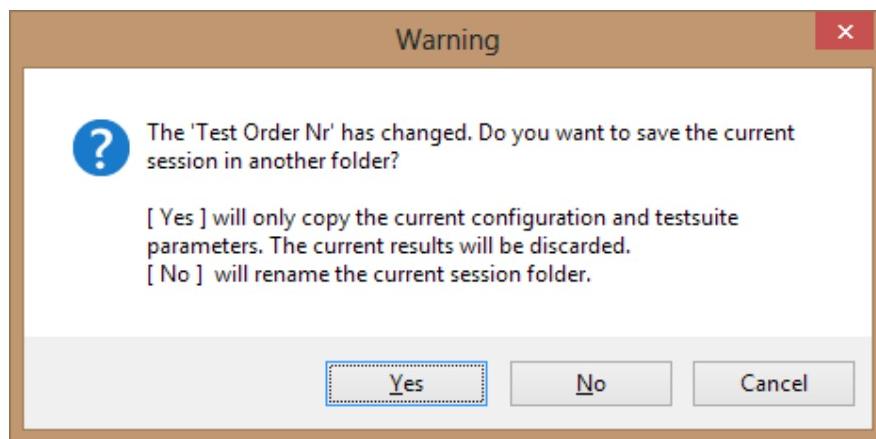


Figure 5.20

More information on the file/folder structure of sessions can be found in Section 6.

Additionally, the user may also access the session settings window by:

- 1) Click in the menu bar under: **File → Save → Save Test Session As...**

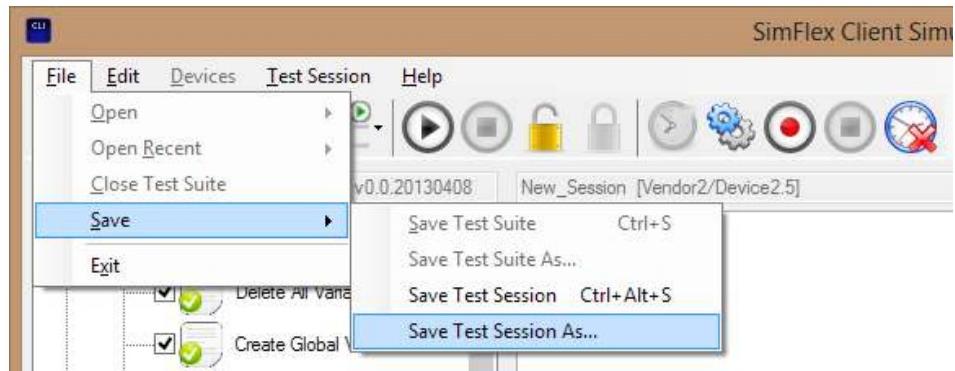


Figure 5.21

- 2) Click in the tool bar button: **Save → Save Test Session As...**

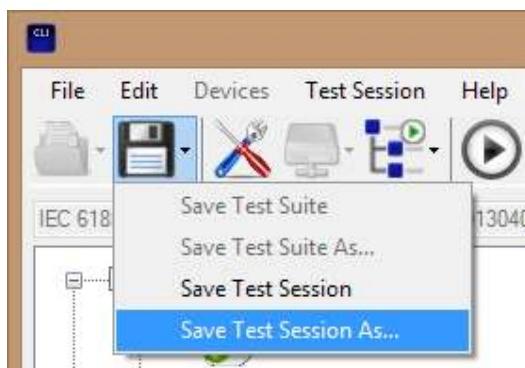


Figure 5.22

An informative window will be opened and after clicking OK on it the Test Session Settings window will be shown.

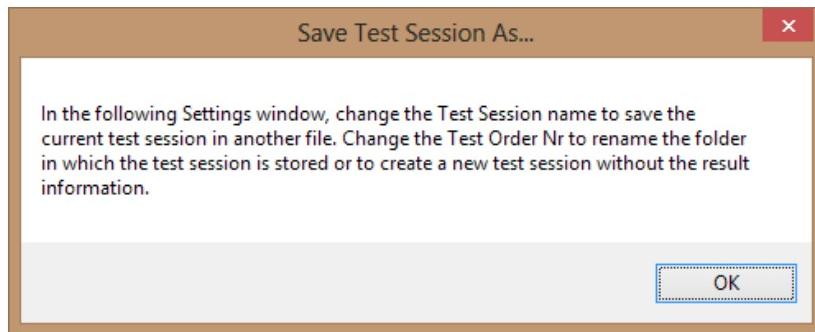
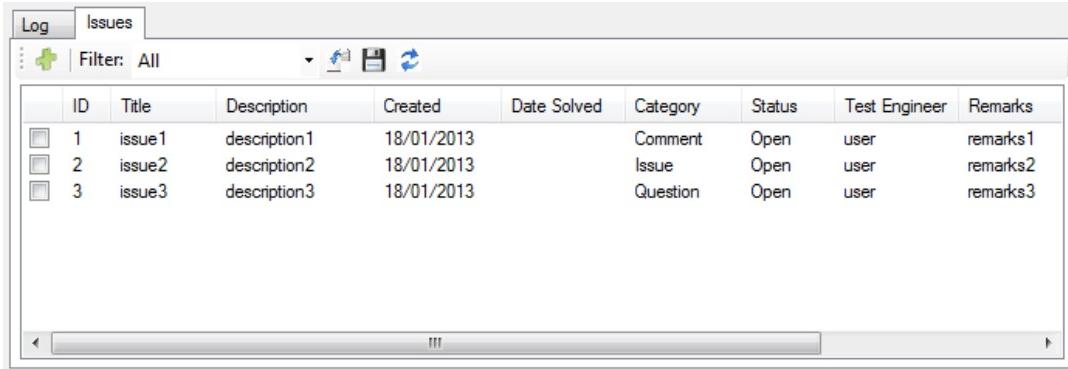


Figure 5.23

## 5.7 Use the issues tab

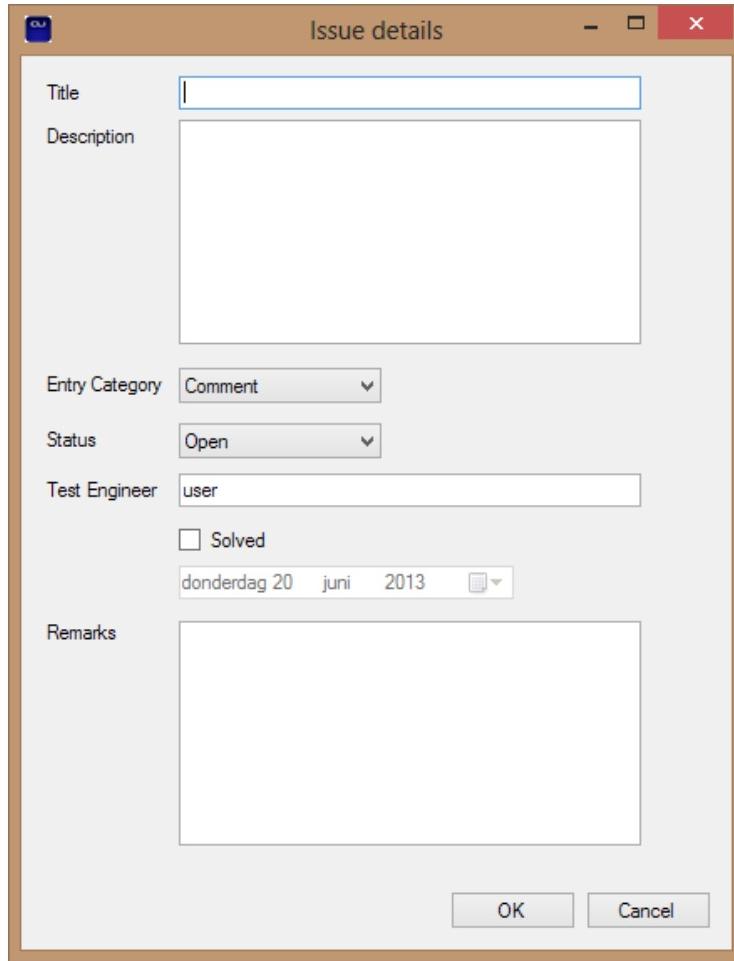
The 'Issues' tab in the lower panel is a small utility to assist the user in writing down and keeping track of issues that appear while testing a device. The tab is only enabled inside sessions.



	ID	Title	Description	Created	Date Solved	Category	Status	Test Engineer	Remarks
<input type="checkbox"/>	1	issue1	description1	18/01/2013		Comment	Open	user	remarks1
<input type="checkbox"/>	2	issue2	description2	18/01/2013		Issue	Open	user	remarks2
<input type="checkbox"/>	3	issue3	description3	18/01/2013		Question	Open	user	remarks3

Figure 5.24

New issues can be added pressing the **Add** button. In the following window the appropriate information can be filled in. Issues can also be edited by double-clicking on them.



The dialog box is titled "Issue details". It contains the following fields:

- Title:** A text input field.
- Description:** A large text area.
- Entry Category:** A dropdown menu set to "Comment".
- Status:** A dropdown menu set to "Open".
- Test Engineer:** A text input field set to "user".
- Solved:** A checkbox that is unchecked.
- Date:** A date picker showing "donderdag 20 juni 2013".
- Remarks:** A large text area.
- Buttons:** "OK" and "Cancel" at the bottom right.

Figure 5.25

The issues can be filtered by category, can be imported from other issue files, can be manually saved and refreshed by using the buttons in the top of the tab.

## 6 Advanced use

### 6.1 Global settings.

Section 2 described the way to access the [ Globals ] tab and that it is used for configuration of the global settings or variables used through the *Client Simulator*.

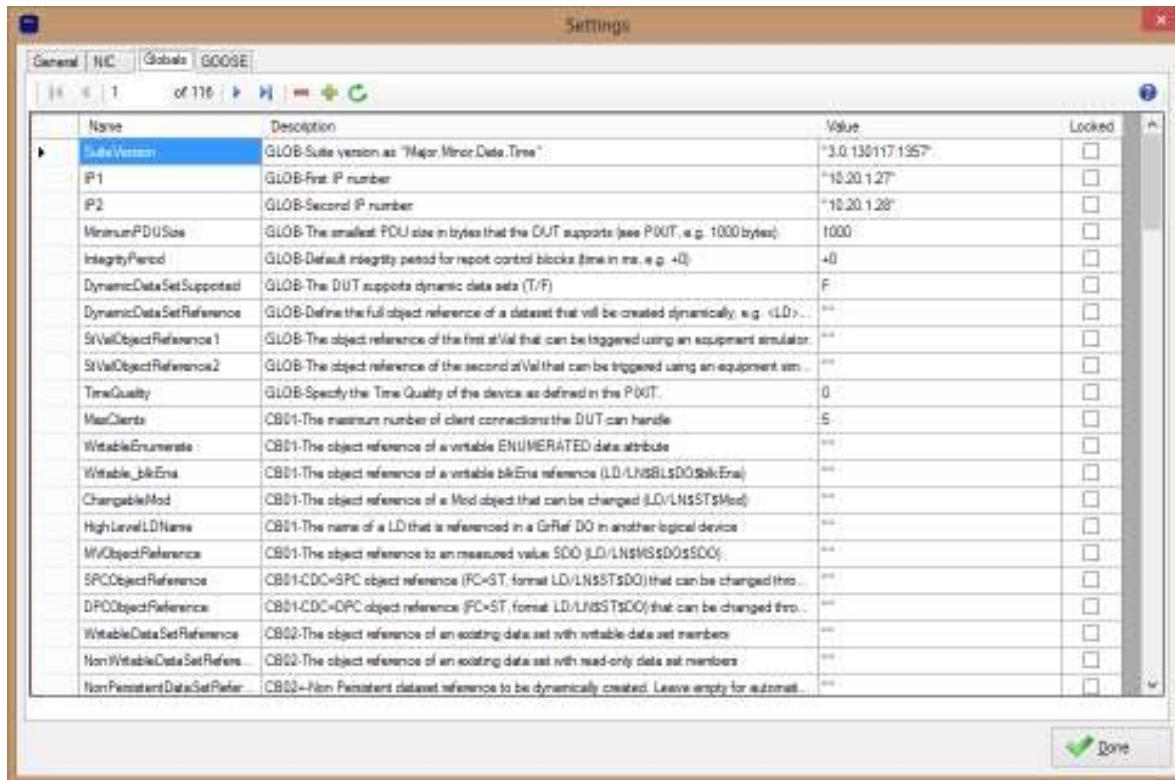


Figure 6.01 Settings window, Globals tab

The tab has a small toolbar used (from left to right) for navigation, remove variable, add new variable, refresh variable and get help.



Figure 6.02 Toolbar in Globals tab

#### 6.1.1 Columns in the Global Settings Tab

**Name:** This column is listing the Global variable definition that is used in the test scripts. Variables in the global settings can be invoked/used in the test scripts. Once assigned the name cannot be changed and also the Locked flag is off. (See below)

**Description:** This Column describe briefly about the specific global variable and how it is configured.

**Value:** This column has to have a value to the specific global variable and cannot be left empty. Value type declaration will be explained below.

**Locked:** If this feature is ticked, all the columns are locked and the user cannot edit the related rows. This will also prevent scripts from changing the value by accident.

### 6.1.2 Global variable format

This part describes the types available for the Global variable and how it is 'serialized' into a text string.

#### BOOLEAN

BOOLEAN type is declared either with T or F.

=> T, F

(Note: The texts True and False will be converted to T and F.)

#### INTEGER

INTEGER type is explicitly declared with numerical int the column

=> 1234567890, -1234567890

(Note: Do not use the '+' sign, see UNSIGNED INTEGER.)

#### UNSIGNED INTEGER

UNSIGNED INTEGER type is explicitly declared with numerical in the column

=> +1234567890

(Note: The '+' sign forces it to be an unsigned integer.)

#### STRING

STRING type is declared with "".

=> "this is how it is declared."

(The double quotes are \*not\* optional and must be included.)

#### BITSTRING

BITSTRING type is declared as a list of zeros and ones within square brackets.

=> [010011...]

#### STRUCTURES AND LISTS

STRUCTURE and LIST types are declared as a comma-separated list of other types within curly brackets.

=> {"first element",2,+3,{"nested","structure"}}

## 6.2 More on Scripts

## SimFlex™ Client Simulator Edition 2 – User Manual

Section 4 described how to run test scripts. A script can be opened by double-clicking on it.

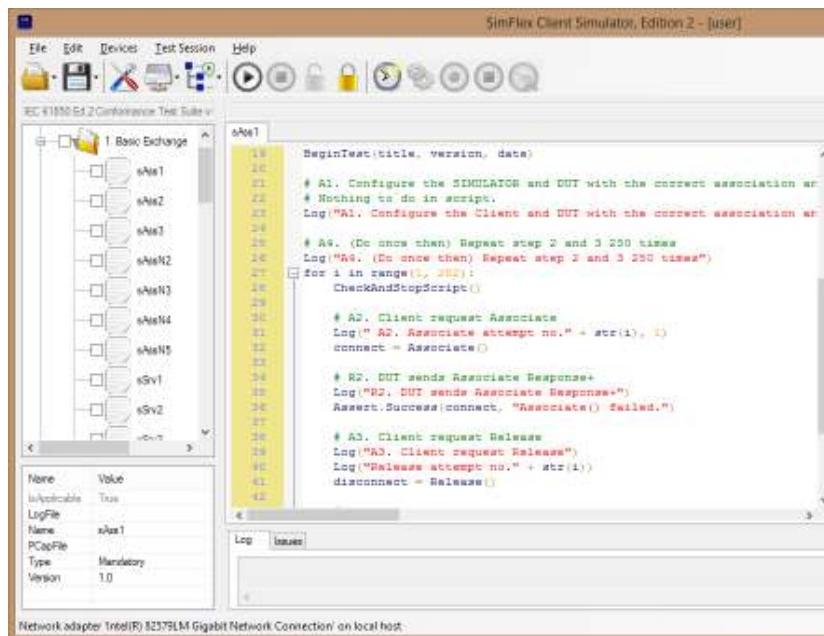


Figure 6.03 Opening a test script

By default, test suites or sessions are opened in read-only mode. That is, the script can be examined by the user but not modified. Do so by scrolling down and up like in a traditional text editor.

In order to modify the content of the script click on the **[ Unlock ]** button:

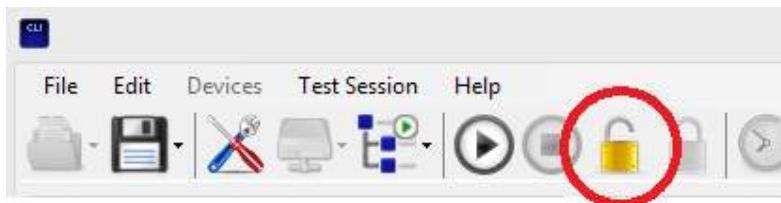


Figure 6.04 Unlock test script

After editing the script it is recommended to protect it against accidental modification by clicking on the **[ Lock ]** button.

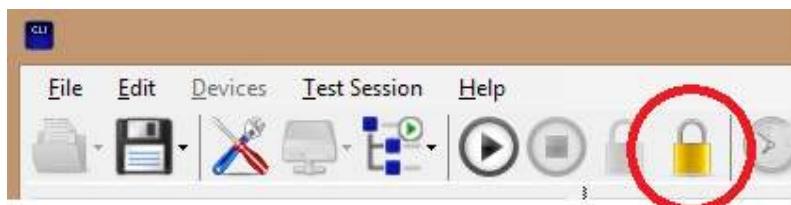


Figure 6.05 Lock test script

### 6.2.1 IEC 61850 and UCA test procedures

IEC 61850-10 defines the methods and abstract test cases for conformance testing of devices used in substation automation systems. It also defines the metrics to be measured within devices according to the requirements defined in IEC 61850-5. Part 10 of the IEC 61850 standard specifies the basic types of tests required to label a device “IEC 61850 conformant” but it leaves details of the tests open to interpretation.

The Utility Communications Architecture International Users’ Group (UCA IUG) formed a Testing Committee to review the collective needs of device users and device providers for Conformance testing. The Testing Committee’s mandate is to create a conformance test system compliant to all parts of IEC 61850 which would meet the needs of users.

The testing committee has created three major documents as the core of the test system: IEC 61850 Conformance Testing:

- The Accreditation Program. This document specifies what conditions an organization must meet before it can claim to be a UCA IUG-recognized (accredited) IEC 61850 conformance tester.
- Quality Assurance Procedures. This document specifies the rules users and conformance testers must follow to ensure that interoperability issues discovered in real systems are prevented by improved wording in future versions of the specification.
- Test Procedures. This document contains detailed step-by-step rules for conducting conformance tests. These rules encode tests for not only normal operating conditions (positive tests) but also as many error conditions (negative tests) as possible. These test procedures reference the conformance testing specified in IEC 61850 Part 10.

The test suite provided with the *SimFlex™ Client Simulator* implements the test procedures described in “*Conformance Test Procedures for Server Devices with IEC 61850-8-1 Edition 2 interface. Revision 1.0*”, specifically in annex “*A4. Mapping of ACSI models and services (IEC 61850-7-2 and applicable SCSM)*”.

### 6.2.2 Script structure and language

The *SimFlex™ Client Simulator* uses Python as the programming language for the test cases because it is a high level programming language, easily understandable and configurable. The user must be familiar with this language’s syntax in order to edit the scripts.

The scripts do not expose the whole Python language; they only use a small subset that can be learnt in minutes by anyone with any previous experience with some kind of computer programming (Matlab, Excel, etc). Many books and tutorial exist covering the basic Python syntax.

Once familiar with the syntax the structure of the scripts will be clear to the user. All of them start with a (optional) declaration of global variables used in the script. Then some kind of initialization follows. The test procedure begins after the *BeginTest()* call. In the script body several function calls, variable assignations, execution control blocks and loops are performed. Finally, the test procedure ends with the *EndTest()* call. Note that the script body tries to follow the test procedure as closely as possible in a literal sense.

If *SimFlex™ Protocol Analyzer* is installed, the test can be extended with analyzes. To implement this, add “*FinalResult = Analyze()*” to the script. This call must be inserted after “*EndTest()*” (because the

capture file has to be closed and that is done in “*EndTest()*”. If *SimFlex™ Protocol Analyzer* is not installed, *Analyze()* will return *Testresult.PASS*.

### 6.2.3 Function calls

In addition to Python syntax and the use of global variables, the *SimFlex™ Client Simulator* exposes an API to interact with the DUT. The API can be inspected in the online help by clicking in the menu **Help → SimFlex CS Python API**.

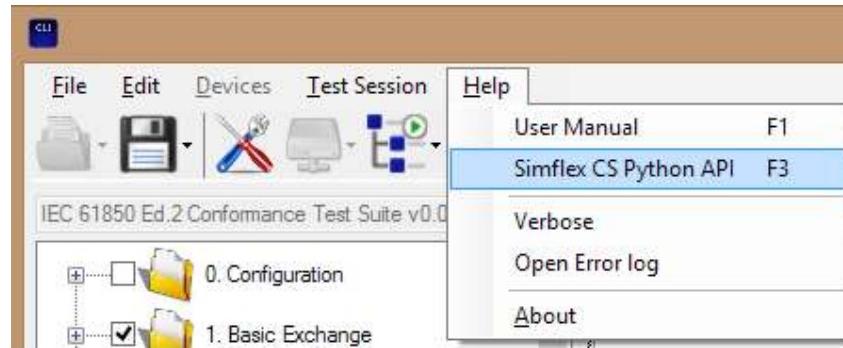


Figure 6.06 SimFlex CS Python API menu

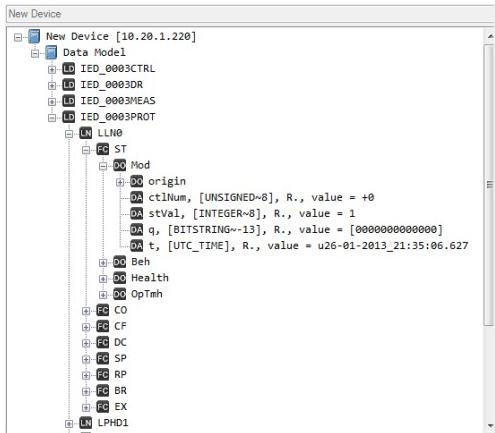
A windows help file will be opened with the aforementioned API documentation:



Figure 6.07 SimFlex CS Python API Docs

## 6.3 How to use the data model

Section 3 describes the procedure to build the DUT’s data model. After doing so, the data model will be shown in the right panel as shown in the next figure:



**Figure 6.08 Data model tree**

### 6.3.1 Tree hierarchy

LD = Logical Device

LN = Logical Node

FC = Functional Constraint

DO = Data Object

DA = Data Attributes

### 6.3.2 How to read the values in the DUT data model (MMSData values)

All values from and to a DUT are communicated using the MMS protocol (ISO 9506), MMS for short, as described in IEC 61850-6. Values shown in the DUT data model are restricted to the format described within the MMS protocol.

The values displayed in the DUT data model do not only contain the MMSData type but also includes some length information, a flag indicating if the value is ‘Readable’, ‘Writable’ or both and the actual value retrieved from the DUT.

The normal format for an item with a value to is in the data model is:

**Name, [MMSData type], Readable/Writable flag, value.**

Example:

DUT\_NAME [DUT IP address]

Data Model

LD name

FC ST

DO Mod

DA stVal, [INTEGER^8], R., value = 1

DA q, [BITSTRING~13], R., value = [00000000000000]

DA t, [UTC\_TIME], R., value = u23-11-2012\_17:05:57.720

### 6.3.3 MMSData type and length information:

The length information with the MMSData type is according to IEC 61850-8, paragraph 8.1, table 11, mapping of ACSI data types.

Most MMSData types with a fixed length (such as UTC\_TIME) have no length information, with the exception of BOOLEAN, INTEGER and UNSIGNED which have length information of one digit (8 for 8 bits, 16 for 16 bits etc.). String based types (such as VISIBLE\_STRING) have length information that can be negative indicating a Maximum length (the string may be shorter than the length but only if the length information is negative.). And FLOATINGPOINT values have two length information values, one to indicate the total length (in bits) and the second to indicate the length of the exponent (in bits).

#### Readable/Writable:

At maximum two characters indicate the data attribute readable/writable information. The possible characters are:

<u>Character</u>	<u>Meaning</u>
-	The Readable/Writable state is unknown.
.	The Readable/Writable state is untested.
R	The data attribute is Readable
W	The data attribute is Writable
w	The data attribute should be writable but is not available at the moment for writing to.

The most common combinations are:

<u>Combination</u>	<u>Meaning</u>
-	The data attribute Readable/Writable state is unknown.
.	The data attribute Readable/Writable state is untested.
R.	The data attribute is Readable but not tested for Writable.
R-	The data attribute is Readable but not Writable.
Rw	The data attribute is Readable and should be Writable but was not available for writing at the moment of testing.
RW	The data attribute is Readable and Writable.

## 6.4 More on configuration files

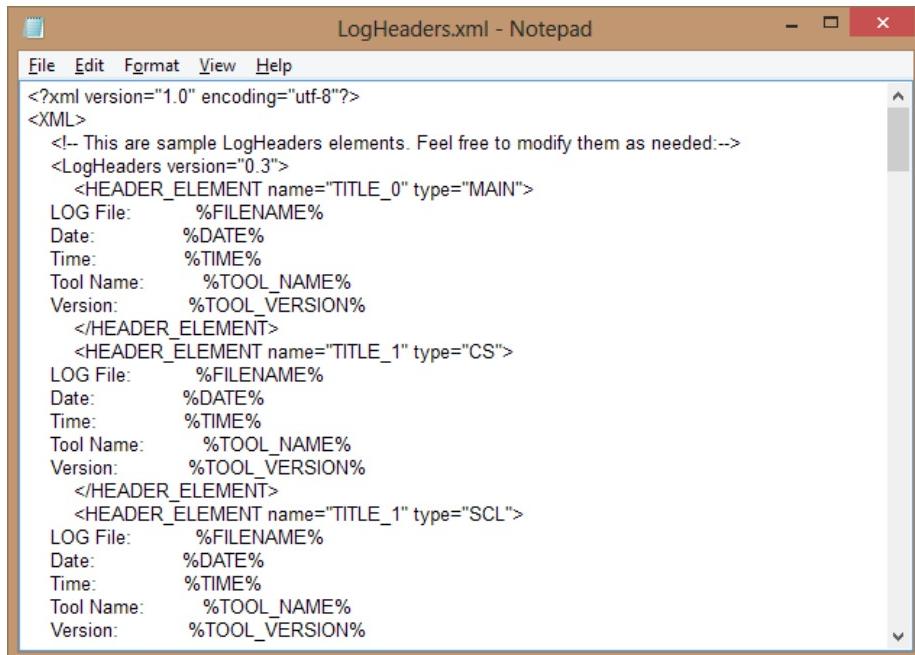
Several files can be edited to alter some of the behavior of the *Client Simulator*.

### 6.4.1 Log file customization

As explained in Section 4, the log text generated for each test script is stored in disk. Depending if the test is run in a session or not it is stored under the ‘*Application base directory*’ folder in:

- *Session (not recording)*: ‘.\Sessions\%VENDOR%\%DEVICE%.%NR%\Pre-Test\’
- *Session (recording)*: ‘.\Sessions\%VENDOR%\%DEVICE%.%NR%\Test\’
- *No Session*: ‘.\TestLab\TestBenches\%TESTBENCH%\Pre-Test\’

The log file has some headers and footers at the beginning and ending. This header and footer can be customized by the user. Log file customization is done by modifying the ‘.\TestLab\LogHeaders.xml’ file.



**Figure 6.09 LogHeaders.xml**

Two kinds of XML elements can be added to this file: HEADER and FOOTER elements:

- **HEADER\_ELEMENTS** are used in log files generated by test scripts. They are displayed before the logging itself in the order found in this file.
- **FOOTER\_ELEMENTS** are used in log files generated by test scripts. They are displayed after anything else in the order found in this file.

Each element can be of either one of three different types: MAIN, CS and SCL.

- **MAIN** types are used as a title for program log files. They are displayed before anything else in the order found in this file.
- **CS** types are used as a header for script log files in the *SimFlex™ Client Simulator*. They are displayed before anything else in the order found in this file.

- **SCL** types are used as a header for script log files in the *SimFlex™ SCL Checker*. They are displayed before anything else in the order found in this file.

Variables in the form of %VAR% can be used inside the element inner text. Two kinds of variables can be used: default variables defined in the *SimFlex™* tool and environment variables defined by the user or the OS.

**Note 1:** Environment variables will be checked before the default *SimFlex™* variables, thus environment variables override the default values. Take this into account when creating new environment variables.

**Note 2:** Some default variables are not applicable at some points, e.g. session information is only available while running a session. When this is the case they will be translated as "NOT-APPLICABLE".

**Note 3:** If a variable, other than the default *SimFlex™* variables, is used and it is not defined as an environment variable, it will be translated as "UNDEFINED".

**Note 4:** The variable %TSCRIPT\_STATUS% may have different values in the title/header and in the footer. The former will be the status of the script \*before\* running the script (i.e. the former status of the script), whereas the later will be the status of the script after running it.

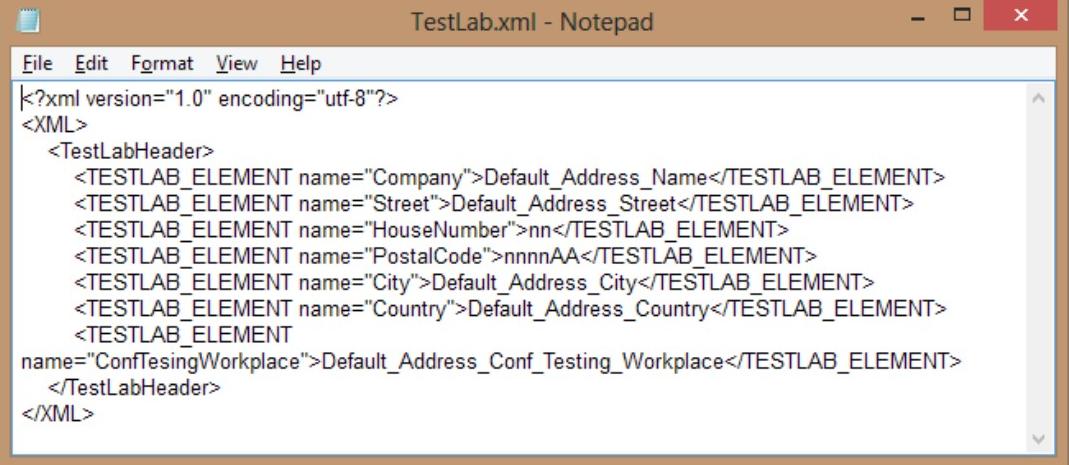
This is a list of the default *SimFlex™* variables available:

%FILENAME%  
%DATE%  
%TIME%  
%TOOL\_NAME%  
%TOOL\_VERSION%  
%TESTER%  
%TESTBENCH%  
%TSUITE\_NAME%  
%TSUITE\_VERSION%  
%TSCRIPT\_NAME%  
%TSCRIPT\_VERSION%  
%TSCRIPT\_STATUS%  
%DEVICE\_NAME%  
%DEVICE\_VENDOR%  
%DEVICE\_VERSION%

%DEVICE\_COMMENTS%  
%DEVICE\_IPADDRESS%  
%DEVICE\_SCLFILE%  
%SESSION\_NAME%  
%SESSION\_ORDERNR%  
%SESSION\_SETUP%  
%SESSION\_VERSION%  
%SESSION\_COMMENTS%  
%TESTLAB\_COMPANY%  
%TESTLAB\_STREET%  
%TESTLAB\_HOUSENUMBER%  
%TESTLAB\_POSTALCODE%  
%TESTLAB\_CITY%  
%TESTLAB\_COUNTRY%  
%TESTLAB\_CTWORKPLACE%  
%CUSTOMER\_COMPANY%  
%CUSTOMER\_STREET%  
%CUSTOMER\_HOUSENUMBER%  
%CUSTOMER\_POSTALCODE%  
%CUSTOMER\_CITY%  
%CUSTOMER\_COUNTRY%  
%CUSTOMER\_CTWORKPLACE%

#### 6.4.2 Test Lab information

Every time a test session is created the Test Lab information must be entered. The default Test Lab information can be modified by opening and editing an xml file under the '*Application base directory*' folder in '*.\TestLab\TestLab.xml*'



```

<?xml version="1.0" encoding="utf-8"?>
<XML>
  <TestLabHeader>
    <TESTLAB_ELEMENT name="Company">Default_Address_Name</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT name="Street">Default_Address_Street</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT name="HouseNumber">nn</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT name="PostalCode">nnnnAA</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT name="City">Default_Address_City</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT name="Country">Default_Address_Country</TESTLAB_ELEMENT>
    <TESTLAB_ELEMENT
      name="ConfTestingWorkplace">Default_Address_Conf_Testing_Workplace</TESTLAB_ELEMENT>
  </TestLabHeader>
</XML>

```

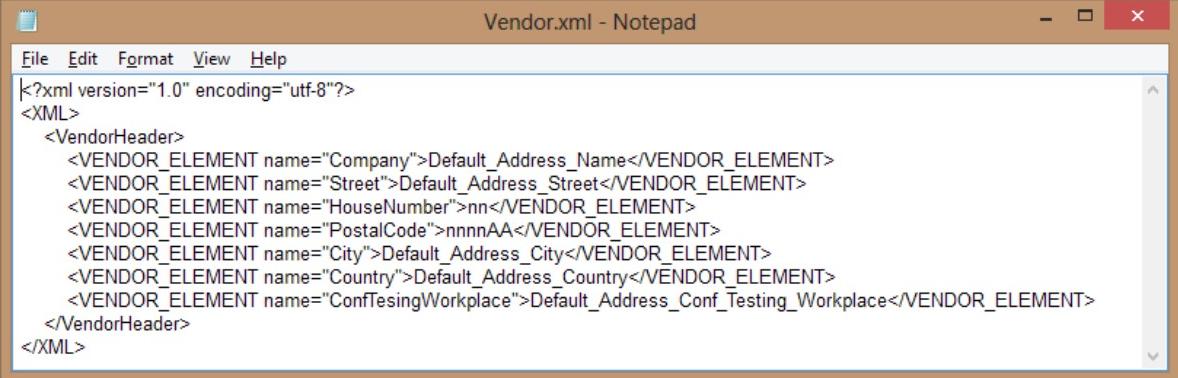
**Figure 6.10 TestLab.xml**

Introduce the company's details under the *Company*, *Street*, *HouseNumber*, *PostalCode*, *City*, *Country* and *ConfTestingWorkplace* attributes.

Every new session created will include the modified information by default.

#### 6.4.3 Vendor information

As with the Test Lab information, every time a test session is created the Vendor information must be entered. Once a vendor has been introduced in the session's folder (by creating the first session for a device from that vendor) an xml file will be created under the '*Application base directory*' folder in *.\Sessions%\%VENDOR%\Vendor.xml*'.



```

<?xml version="1.0" encoding="utf-8"?>
<XML>
  <VendorHeader>
    <VENDOR_ELEMENT name="Company">Default_Address_Name</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="Street">Default_Address_Street</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="HouseNumber">nn</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="PostalCode">nnnnAA</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="City">Default_Address_City</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="Country">Default_Address_Country</VENDOR_ELEMENT>
    <VENDOR_ELEMENT name="ConfTestingWorkplace">Default_Address_Conf_Testing_Workplace</VENDOR_ELEMENT>
  </VendorHeader>
</XML>

```

**Figure 6.11 Vendor.xml**

Introduce the vendor's details under the *Company*, *Street*, *HouseNumber*, *PostalCode*, *City*, *Country* and *ConfTestingWorkplace* attributes.

Every new session created for one of that vendor's devices will include the modified information by default.

## 7 Frequently Asked Questions (FAQ)

Some questions do reoccur and become frequently asked (that is, more than once). Check this section before you contact GridClone for support on your question.

### 7.1 General

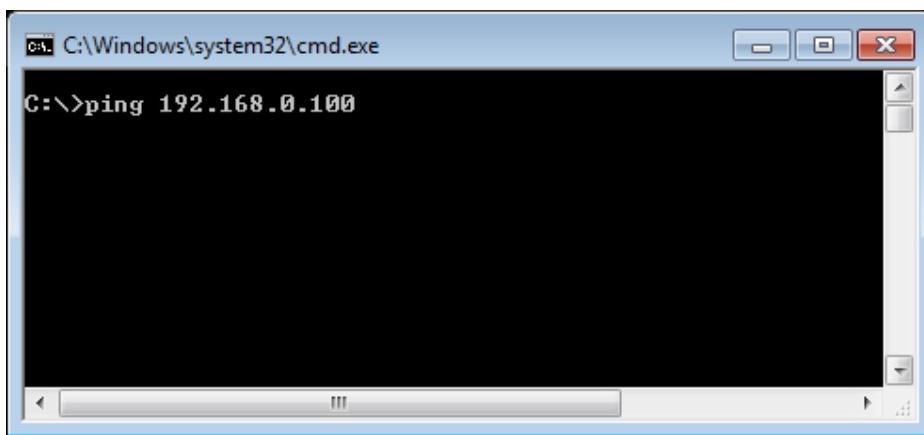
#### 7.1.1 How to install the SimFlex™ Client Simulator?

Read the *Getting started* document that comes with the program for a detailed description of the installation procedure. If you do not have **administrator rights** on your PC, please contact your IT department.

#### 7.1.2 How can I check if my device is available on the network?

See that your DUT is turned on and connected to the same LAN as your *Client Simulator*. Find out the IP address of the DUT.

A simple test to check if the DUT is connected and within range of the *Client Simulator* is to open a Windows Command Window (cmd.exe) and typing the following instruction at the caret:



The IP address behind the *ping* command shall be the IP address of the DUT. In case the DUT is visible it will react with positive responds.

Next, be sure that you selected the appropriate **network interface** in the [ **General** ] and [ **NIC** ] tabs in the Configuration Dialog of the *Client Simulator*.

After that be sure that you created a DUT in the [ **Devices** ] tab in the Configuration Dialog of the *Client Simulator*. Check if the IP address is the same as the IP address of your DUT.

Try to retrieve the date model of the DUT.

#### 7.1.3 The Client Simulator does not start or complains about the license

In the system tray check if the CodeMeter service is running (clicking on the small system-tray arrow).

If the CodeMeter is not visible, it could be that it is not running, or that the service **\*is\*** running, but the CodeMeter program is not visible.

- a) In the first case: re-install the *Client Simulator* and make sure the WIBU software installs correctly.
- b) In the second case: press Windows Start button and locate the CodeMeter installation. Then start the “CodeMeter Control Center” manually.

Once the CodeMeter Control Center is started, open it (from the system tray) and check that there is a license shown in the left part of the dialog. Make sure you have inserted the USB license key in one of the available USB ports!

In case you are running a trial version of the SimFlex™ Client Simulator:

If there is no license shown, select **File→Import License** from the menu and locate the “\*.lic” file in the *Client Simulator* folder (In the folder where you installed the program, e.g. C:\Program Files\GridClone\...). Note that the CodeMeter Control Center might have a filter like “\*.WibuCmRaU”. Change it to “\*.\*” or “\*.lic” to make the license file visible.

Once the license file has been selected and opened it should be visible in the CodeMeter Control Center.

To be sure the license is activated: stop and start the CodeMeter Control Center service by pressing **Process→Stop CodeMeter Service** and then **Process→Start CodeMeter Service**.

Finally re-start the *Client Simulator*. If the problem persists please contact GridClone for support.

#### 7.1.4 Missing Module ‘OS’

After the installation of the SimFlex™ Client Simulator, one of the first actions that a user does is creating the global variables. This is done by enabling in the test suite the group “0. Configuration” and the script “Create Global Variables”. See the excerpt of the test suite below.



In some cases the following error might occur when the global variables are being created:



The reason can be that IronPython was not installed in its default location.

Solution: add the following environment variable (a system variable, not a user variable) in Windows and retry to create the global variables in the SimFlex™ Client Simulator (restart of the program first).

*Set SimflexipyLib=<path to IronPython lib folder>*

*Example:*

*Set Simflex\pyLib=C:\Program Files (x86)\IronPython 2.7.1\Lib*

## 7.2 Test suites.

### 7.2.1 How to copy a test suite from a session?

Read also:

- Part 4.1 ‘Open a test suite.’
- Part 5.6 ‘Modify and copy test sessions’.

Note that test sessions created with SimFlex™ CS Edition 1 cannot be loaded with SimFlex™ CS Edition 2 due to differences in model and test structures.

The test suite can be found in one of the following places, depending on the use of the suite.

#### ‘Used outside of a session’:

%APPLICATION\_BASE%\TestLab\TestSuites\\*.session

The test suite is not bound to a test session and is (most likely) not configured. It can be used as a base suite to construct a test session.

**NOTE:** Initially the folder “%APPLICATION\_BASE%\TestLab\TestSuites” might not contain the IEC 61850 Edition 2 test suite. In that case the test suite is located in the installation folder of the SimFlex CS, e.g. “C:\Program Files (x86)\GridClone\SimFlex Client Simulator\Testsuites” (32bit version of Windows) or “C:\Program Files (x86)\GridClone\SimFlex Client Simulator\Testsuites” (64bits version of Windows) for the default installation of the program.

#### ‘Used in a session’:

%APPLICATION\_BASE%\Session%\VENDOR%\DEVICE%[.TEST\_NR]\TestSuites\\*.tsf

The test suite is bound to a test session, the details are in the .tsf file, including all scripts and configuration details.

The preferred way for copying sessions is through the Test session settings form and changing the field [Test order number] (See 5.6 ‘Modify and copy test sessions’). It can be copied to another session area and used ‘as is’.

%APPLICATION\_BASE% = ‘C:\ProgramData\GridClone\SimFlex’

By default. It can be changed in the configuration (see Edit → Settings, tab [ General ] field ‘Application base directory’ for the actual setting).

%VENDOR%, %DEVICE% and %TEST\_NR% is data from the actual session referring to. The %TEST\_NR% is optional.

## 8 Additional Information

In this section you will find information on additional topics.

### 8.1 File name conventions

Any \*.txt file is a test result file and has the same content as the log frame.

Any \*.log file is an error log file for errors related to the program and not to the test.

**sAss1-2013-01-27\_09.01.02-CS.txt** is the test result of test case sAss1 generated by the *Client Simulator* on January 27<sup>th</sup>, 2013 at 09:01 (local time).

**2013-01-07\_09.28.43-SimFlexError.log** is an error log for the same run. This file is located in the “Application base directory”.

## 8.2 Used abbreviations

DUT Device Under Test.

IEC The International Electrotechnical Commission, managing international standards. See: <http://www.iec.ch/>

IEC 61850 The IEC 61850:2007 standard, revision A and B

IED Intelligent Electronic Device

UCA The UCA® International Users Group. See: <http://www.ucaiug.org>

## 8.3 Referenced documents

IEC 61850 IEC 61850:2007 revision A and B

MMS ISO 9506:2003 (as used by IEC 61850-8)

Getting started.pdf

## 8.4 Referenced tools

WinPcap For capturing network traffic, see: <http://www.winpcap.org/>

WireShark For analyzing network traffic, see: <http://www.wireshark.org/>

## 8.5 Document Version History

The table below describes the version history of this document.

Version	Date	By	Subject
1.0	April 15, 2013	EM	Initial version
1.1	June 19, 2013	EM	General update before release.
1.2	December 16, 2013	WvE	SimFlex™ Protocol Analyzer integrated